



**TUGAS AKHIR - TE145561**

**PEMBUATAN MODUL KONTROL KECEPATAN MOTOR  
*BRUSHLESS* DC DENGAN MIKROKONTROLER**

Intan Nur Robi Annisa  
NRP 2213039012  
Zaka Perwira  
NRP 2213039043

Dosen Pembimbing  
Ir. Josaphat Pramudijanto, M.Eng.  
Onie Meiyanto, S.Pd.

PROGRAM STUDI D3 TEKNIK ELEKTRO  
Fakultas Teknologi Industri  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016



**TUGAS AKHIR - TE145561**

**PEMBUATAN MODUL KONTROL KECEPATAN MOTOR  
*BRUSHLESS* DC DENGAN MIKROKONTROLER**

Intan Nur Robi Annisa  
NRP 2213039012  
Zaka Perwira  
NRP 2213039043

Dosen Pembimbing  
Ir. Josaphat Pramudijanto, M.Eng.  
Onie Meiyanto, S.Pd.

PROGRAM STUDI D3 TEKNIK ELEKTRO  
Fakultas Teknologi Industri  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016



**FINAL PROJECT- TE145561**

***CONSTRUCTION OF SPEED CONTROL MODULE OF  
BRUSHLESS DC MOTOR USING MICROCONTROLLER***

Intan Nur Robi Annisa  
NRP 2213039012  
Zaka Perwira  
NRP 2213039043

*Supervisor*

Ir. Josaphat Pramudijanto, M.Eng.  
Onie Meiyanto, S.Pd.

*Electrical Engineering D3 Study Program  
Industrial Technology Faculty  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016*



**FINAL PROJECT- TE145561**

***CONSTRUCTION OF SPEED CONTROL MODULE OF  
BRUSHLESS DC MOTOR USING MICROCONTROLLER***

Intan Nur Robi Annisa  
NRP 2213039012  
Zaka Perwira  
NRP 2213039043

*Supervisor*

Ir. Josaphat Pramudijanto, M.Eng.  
Onie Meiyanto, S.Pd.

*Electrical Engineering D3 Study Program  
Industrial Technology Faculty  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016*

## PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**Pembuatan Modul Kontrol Kecepatan Motor *Brushless* DC Dengan Mikrokontroler**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

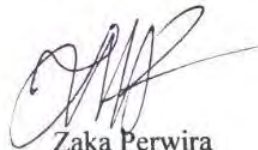
Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juni 2016



Intan Nur Robi Annisa  
NRP 2213039012



Zaka Perwira  
NRP 2213039043

*-- Halaman ini sengaja dikosongkan --*

# **PEMBUATAN MODUL KONTROL KECEPATAN MOTOR BRUSHLESS DC DENGAN MIKROKONTROLER**

## **TUGAS AKHIR**

**Diajukan Guna Memenuhi Sebagian Persyaratan  
Untuk Memperoleh Gelar Ahli Madya Teknik**

**Pada**

**Bidang Studi Elektro Industri  
Program Studi D3 Teknik Elektro  
Fakultas Teknologi Industri  
Institut Teknologi Sepuluh Nopember**

**Menyetujui :  
Dosen Pembimbing,**



**Ir. Josaphat Pramudijanto, M.Eng.**  
**NIP. 19621005 199003 1 003**

**SURABAYA  
JUNI, 2016**

*-- Halaman ini sengaja dikosongkan --*



**PEMBUATAN MODUL KONTROL KECEPATAN MOTOR  
BRUSHLESS DC DENGAN MIKROKONTROLER**

**TUGAS AKHIR**

Diajukan Guna Memenuhi Sebagian Persyaratan  
Untuk Memperoleh Gelar Ahli Madya Teknik

Pada


Bidang Studi Elektro Industri

Program Studi D3 Teknik Elektro

Fakultas Teknologi Industri

Institut Teknologi Sepuluh Nopember

Menyetujui :  
Dosen Pembimbing,



Onie Meiyanto, S.Pd.

NIP. 19850501 201101 1 008

**SURABAYA  
JUNI, 2016**

*-- Halaman ini sengaja dikosongkan --*

**LEMBAR PERNYATAAN  
PERSETUJUAN PUBLIKASI KARYA ILMIAH  
UNTUK KEPENTINGAN AKADEMIS**

Sebagai mahasiswa Institut Teknologi Sepuluh Nopember Surabaya, yang bertanda tangan di bawah ini saya :

Nama : INTAN NUR ROBI A / ZAKA PERWIRA  
Nrp. : 2213 039 012 / 2213 039 043  
Jurusan / Fak : 03 TEKNIK ELEKTRO / FTI  
Alamat kontak : 11 Ketawan Putih Tambak Baru Gg IX NO 10 D, Surabaya  
a. Email : zakapenwira14@gmail.com  
b. Telp HP : 082330265496

Menyatakan bahwa semua data yang saya *upload* di Digital Library ITS merupakan hasil final (revisi terakhir) dari karya ilmiah saya yang sudah disahkan oleh dosen penguji. Apabila dikemudian hari ditemukan ada ketidaksesuaian dengan kenyataan, maka saya bersedia menerima sanksi.

Demi perkembangan ilmu pengetahuan, saya menyetujui untuk memberikan Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalti-Free Right*) kepada Institut Teknologi Sepuluh Nopember Surabaya atas karya ilmiah saya yang berjudul :

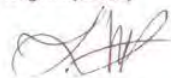
PEMBUATAN MODUL KONTROL KECEPATAN MOTOR BRUSHLESS DC DENGAN MIKROKONTROLER

Dengan Hak Bebas Royalti Non-Eksklusif ini, Institut Teknologi Sepuluh Nopember Surabaya berhak menyimpan, mengalih-media format-kan, mengelolanya dalam bentuk pangkalan data (*database*), mendistribusikannya, dan menampilkan mempublikasikannya di internet atau media lain untuk kepentingan akademik tanpa meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis pencipta. Saya bersedia menanggung secara pribadi, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini tanpa melibatkan pihak Institut Teknologi Sepuluh Nopember Surabaya.

Demikian pernyataan ini saya buat dengan sebenarnya.



Dibuat di : Surabaya  
Pada tanggal : 27 JUNI 2016  
Yang menyatakan,

  
ZAKA PERWIRA  
Nrp. 2213 039 043

**KETERANGAN :**

Tanda tangan pembimbing wajib dibubuhi stempel jurusan.

Form dicetak dan diserahkan di bagian Pengadaan saat mengumpulkan hard copy TA/Tesis/Disertasi.

*-- Halaman ini sengaja dikosongkan --*

## **PEMBUATAN MODUL KONTROL KECEPATAN MOTOR *BRUSHLESS* DC DENGAN MIKROKONTROLER**

**Nama Mahasiswa : Intan Nur Robi Annisa**  
**NRP : 2213 039 012**  
**Nama Mahasiswa : Zaka Perwira**  
**NRP : 2213 039 043**  
**Dosen Pembimbing 1 : Ir. Josaphat Pramudijanto, M.Eng.**  
**NIP : 19621005 199003 1 003**  
**Dosen Pembimbing 2 : Onie Meiyanto, S.Pd.**  
**NIP : 19850501 201101 1 008**

### **ABSTRAK**

Dalam proses pembelajaran, untuk mengetahui kecepatan dari motor *brushless* DC diperlukan sebuah alat ukur yang bisa memberikan data yang tepat. Selain itu, untuk mengontrol kecepatan dari motor *brushless* DC diperlukan pula sebuah modul kontrol. Namun, modul khusus untuk kontrol kecepatan motor *brushless* DC tidak mudah untuk dibuat. Sementara itu masih banyak mahasiswa yang belum tahu bagaimana cara mengontrol motor *brushless* DC dengan benar.

Oleh karena itu untuk mempermudah proses pembelajaran, diperlukan suatu modul yang praktis dan efisien. Sehingga muncul ide untuk membuat suatu modul yang digunakan untuk mengontrol kecepatan motor *brushless* DC dengan mikrokontroler. Modul ini terdiri dari 4 komponen utama, yaitu arduino sebagai mikrokontroler, motor *brushless* DC, beban (rem magnetik), dan sensor kecepatan. Modul ini kemudian dioperasikan melalui *software* LabView dengan metode kontroler PID dan menggunakan komunikasi jaringan *wireless* Ad Hoc.

Pada tugas akhir ini, dihasilkan bahwa metode kontrol dari motor *brushless* DC dengan kontroler PID *close loop* melalui pengujian perubahan kecepatan dari 1000 RPM sampai 5000 RPM menghasilkan *settling time* yang lebih cepat daripada sebelum menggunakan kontroler PID yaitu sebesar 83 ms pada pengereman 0%, 30 ms pada pengereman 25%, dan 3 ms pada pengereman 50%.

**Kata Kunci : Motor *Brushless* DC, Rem Magnetik, LabView, dan PID.**

*-- Halaman ini sengaja dikosongkan --*

## **CONSTRUCTION OF SPEED CONTROL MODULE OF BRUSHLESS DC MOTOR USING MICROCONTROLLER**

***Student's Name*** : Intan Nur Robi Annisa  
***Registration Number*** : 2213 039 012  
***Student's Name*** : Zaka Perwira  
***Registration Number*** : 2213 039 043  
***Supervisor 1*** : Ir. Josaphat Pramudijanto, M.Eng.  
***ID*** : 19621005 199003 1 003  
***Supervisor 2*** : Onie Meiyanto, S.Pd.  
***ID*** : 19850501 201101 1 008

### **ABSTRACT**

*In the process of learning, to know how is the speed of brushless DC motor, it is required some measurement tools that can provide the right data. In addition, to control the speed of brushless DC motor is also required a control module. However, a special module for brushless DC motor speed control is not easy to be made. While, there are many students who do not yet know how to control brushless DC motor correctly.*

*Therefore, to facilitate the learning process, we need a module that is practical and efficient. So the idea to create a module that is used to control the speed of brushless DC motor with a microcontroller is come. This module consists of four main components, namely arduino as a microcontroller, brushless DC motors, load (magnetic brake), and a speed sensor. The module is then operated via LabView software with PID controller method and using wireless Ad Hoc network communications.*

*In this thesis, the result of the method of a brushless DC motor with closed loop PID controller through testing changes in speed from 1000 RPM to 5000 RPM generate settling time faster than before using the PID controller that is equal to 83 ms at braking 0%, 30 ms at braking 25%, and 3 ms at braking 50%.*

**Keywords :** ***Brushless DC Motor, Magnetic Brake, LabView, and PID.***

*-- Halaman ini sengaja dikosongkan --*



## KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa karena atas limpahan rahmat dan karunia-Nya, penulis dapat menyelesaikan tugas akhir ini dengan judul :

### **“PEMBUATAN MODUL KONTROL KECEPATAN MOTOR *BRUSHLESS* DC DENGAN MIKROKONTROLER”**

Tugas Akhir ini merupakan sebagian syarat untuk menyelesaikan mata kuliah dan memperoleh nilai pada Tugas Akhir.

Dengan selesainya Tugas Akhir ini penulis menyampaikan terima kasih sebesar-besarnya kepada:

1. Orang Tua atas limpahan doa, kasih sayang, dukungan dan dorongan baik berupa moril atau materil bagi penulis.
2. Bapak Joko Susila, Ir., MT. selaku Ketua Program D3 Teknik Elektro Bidang Studi Teknik Listrik, FTI-ITS Surabaya.
3. Bapak Ir. Josaphat Pramudijanto, M.Eng. dan bapak Onie Meiyanto, S.Pd. selaku Dosen Pembimbing.
4. Bapak Slamet Budiprayitno, ST., MT., bapak Moh. Abdul Hady, ST., MT., dan ibu Yunafi'atul Aniroh, ST., M.Sc. selaku Dosen Penguji.
5. Seluruh staf pengajar dan administrasi Prodi D3 Teknik Elektro FTI-ITS.
6. Seluruh Mahasiswa D3 Teknik Elektro khususnya angkatan 2013.
7. Semua pihak yang telah banyak membantu untuk menyelesaikan Tugas Akhir ini yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari masih banyak kekurangan dalam Tugas Akhir ini. Kritik dan saran untuk perbaikan tugas ini sangat diperlukan. Akhir kata semoga tugas ini dapat bermanfaat bagi kita semua.

Surabaya, Juni 2016

Penulis

*-- Halaman ini sengaja dikosongkan --*

# DAFTAR ISI

<b>HALAMAN JUDUL .....</b>	<b>i</b>
<b>LEMBAR PENGESAHAN .....</b>	<b>iii</b>
<b>ABSTRAK .....</b>	<b>ix</b>
<b><i>ABSTRACT</i> .....</b>	<b>xi</b>
<b>KATA PENGANTAR.....</b>	<b>xiii</b>
<b>DAFTAR ISI.....</b>	<b>xv</b>
<b>DAFTAR GAMBAR.....</b>	<b>xvii</b>
<b>DAFTAR TABEL .....</b>	<b>xix</b>
 <b>BAB I PENDAHULUAN .....</b>	 <b>1</b>
1.1 Latar Belakang .....	1
1.2 Permasalahan.....	1
1.3 Batasan Masalah.....	1
1.4 Tujuan .....	2
1.5 Sistematika Penulisan.....	2
1.6 Relevansi .....	3
 <b>BAB II TEORI PENUNJANG .....</b>	 <b>5</b>
2.1 Motor <i>Brushless</i> DC.....	5
2.1.1 Cara Kerja Motor <i>Brushless</i> DC.....	6
2.1.2 Konstruksi Motor <i>Brushless</i> DC.....	7
2.1.3 ESC ( <i>Electronic Speed Controller</i> ) .....	9
2.2 Sensor <i>Rotary Encoder</i> .....	10
2.3 Pengereman Magnetik.....	10
2.4 Mikrokontroler Arduino MEGA 2560 .....	12
2.5 Jaringan <i>Wireless</i> Ad Hoc .....	14
2.6 LabView.....	15
2.7 Kontroler PID .....	15
2.7.1 Kontroler PID <i>Continous</i> .....	16
 <b>BAB III PERANCANGAN DAN PEMBUATAN ALAT .....</b>	 <b>19</b>
3.1 Blok Fungsional Sistem .....	19
3.2 Setting Jaringan <i>Wireless</i> Ad Hoc.....	20
3.3 Perancangan Perangkat Keras ( <i>Hardware</i> ) .....	24
3.3.1 Perancangan Mekanik.....	24
3.3.1.1 Desain Alat Secara Keseluruhan .....	24
3.3.1.2 Rem Magnetik.....	25
3.3.2 Perancangan Elektrik.....	26

3.3.2.1 Rangkaian Sensor <i>Rotary Encoder</i> .....	26
3.3.2.2 <i>Driver</i> Rem Magnetik .....	27
3.4 Perancangan Perangkat Lunak ( <i>Software</i> ) .....	28
3.4.1 Pemrograman <i>Software</i> Arduino .....	28
3.4.2 Perancangan Perangkat Lunak LabView .....	30
<b>BAB IV PENGUJIAN DAN ANALISA</b> .....	31
4.1 Analisa Program Arduino .....	31
4.2 Pengujian Sensor <i>Rotary Encoder</i> .....	33
4.3 Pengujian Kecepatan Putaran Motor <i>Brushless</i> DC Tanpa Kontroler PID.....	36
4.3.1 Pengujian Kecepatan Motor Tanpa Pengereman.....	36
4.3.2 Pengujian Kecepatan Motor Dengan Setengah Pengereman .....	38
4.3.3 Pengujian Kecepatan Motor Dengan Pengereman Penuh.....	40
4.4 Pengujian <i>Software</i> .....	42
4.4.1 Kontroler PID LabView.....	42
4.4.2 Pengujian Motor <i>Brushless</i> DC Dengan Kontroler PID .....	42
<b>BAB V PENUTUP</b> .....	49
5.1 Kesimpulan .....	49
5.2 Saran.....	49
<b>DAFTAR PUSTAKA</b> .....	51
<b>LAMPIRAN A <i>LISTING</i> PROGRAM</b> .....	A-1
<b>LAMPIRAN B <i>DATASHEET</i></b> .....	B-1
<b>LAMPIRAN C DOKUMENTASI</b> .....	C-1
<b>DAFTAR RIWAYAT HIDUP</b> .....	D-1

## DAFTAR GAMBAR

<b>Gambar 2.1</b>	Salah Satu Contoh BLDC.....	5
<b>Gambar 2.2</b>	Skema Kerja Motor BLDC.....	6
<b>Gambar 2.3</b>	<i>Outer Construction</i> .....	7
<b>Gambar 2.4</b>	<i>Inner Construction</i> .....	7
<b>Gambar 2.5</b>	ESC.....	9
<b>Gambar 2.6</b>	<i>Rotary encoder</i> .....	10
<b>Gambar 2.7</b>	Garis Medan Magnet Di Dalam <i>Solenoida</i> .....	11
<b>Gambar 2.8</b>	Gaya Pengereman Yang Dihasilkan Oleh Arus Melingkar Eddy .....	12
<b>Gambar 2.9</b>	Arduino MEGA 2560 .....	13
<b>Gambar 2.10</b>	<i>Setting Serial Port</i> .....	13
<b>Gambar 2.11</b>	Jaringan <i>Wireless Ad Hoc</i> .....	14
<b>Gambar 2.12</b>	Blok Diagram Kontroler PID Sistem <i>Close Loop</i> .....	16
<b>Gambar 3.1</b>	Blok Diagram Perancangan .....	20
<b>Gambar 3.2</b>	Blok Diagram Sistem <i>Close Loop</i> PID .....	20
<b>Gambar 3.3</b>	<i>Start Screen</i> Windows 10 .....	21
<b>Gambar 3.4</b>	<i>Command Prompt</i> .....	22
<b>Gambar 3.5</b>	SSID Dan Key.....	22
<b>Gambar 3.6</b>	Memulai <i>Wireless Ad Hoc</i> .....	23
<b>Gambar 3.7</b>	<i>Local Area Connection</i> .....	23
<b>Gambar 3.8</b>	Desain Modul Kontrol Kecepatan Motor Brushless DC Dengan Mikrokontroler.....	24
<b>Gambar 3.9</b>	Konfigurasi Rem Magnetik .....	25
<b>Gambar 3.10</b>	Rangkaian <i>Rotary Encoder</i> .....	26
<b>Gambar 3.11</b>	<i>Driver</i> Rem Magnetik.....	27
<b>Gambar 3.12</b>	<i>Flowchart</i> Arduino MEGA 2560.....	29
<b>Gambar 3.16</b>	<i>Flowchart</i> LabView .....	30
<b>Gambar 4.1</b>	Potongan Program Inisialisasi .....	32
<b>Gambar 4.2</b>	Potongan Program Komunikasi Data .....	33
<b>Gambar 4.3</b>	Potongan Program Sensor Kecepatan.....	33
<b>Gambar 4.4</b>	Potongan Program Rem Magnetik.....	33
<b>Gambar 4.5</b>	Linierisasi Sensor <i>Rotary Encoder</i> .....	34
<b>Gambar 4.6</b>	Sinyal Pulsa Ketika <i>Rotary Encoder</i> Membaca Kecepatan 3000 RPM.....	34
<b>Gambar 4.7</b>	Rangkaian Pengujian Sensor <i>Rotary Encoder</i> .....	35
<b>Gambar 4.8</b>	Rangkaian Pengujian Kecepatan Motor (Tanpa Pengereman) .....	36
<b>Gambar 4.9</b>	Hasil Pengujian Kecepatan Motor (Tanpa Pengereman) .....	37

<b>Gambar 4.10</b>	Rangkaian Pengujian Kecepatan Motor (Dengan Setengah Pengereman) .....	38
<b>Gambar 4.11</b>	Hasil Pengujian Kecepatan Motor (Dengan Setengah Pengereman).....	39
<b>Gambar 4.12</b>	Rangkaian Pengujian Kecepatan Motor (Dengan Pengereman Penuh).....	40
<b>Gambar 4.13</b>	Hasil Pengujian Kecepatan Motor (Dengan Pengereman Penuh).....	41
<b>Gambar 4.14</b>	Blok Diagram Kontroler PID LabView .....	42
<b>Gambar 4.15</b>	Kecepatan Motor <i>Brushless</i> DC Pengereman 0% Tanpa Kontroler PID .....	43
<b>Gambar 4.16</b>	Kecepatan Motor <i>Brushless</i> DC Pengereman 25% Tanpa Kontroler PID .....	44
<b>Gambar 4.17</b>	Kecepatan Motor <i>Brushless</i> DC Pengereman 50% Tanpa Kontroler PID .....	44
<b>Gambar 4.18</b>	<i>History</i> Perbandingan Kecepatan Motor <i>Brushless</i> DC Pengereman 25% .....	45
<b>Gambar 4.19</b>	<i>History</i> Perbandingan Kecepatan Motor <i>Brushless</i> DC Pengereman 25% .....	45
<b>Gambar 4.20</b>	Perbandingan Kecepatan Motor <i>Brushless</i> DC Pengereman 0%.....	45
<b>Gambar 4.21</b>	Perbandingan Kecepatan Motor <i>Brushless</i> DC Pengereman 25%.....	47
<b>Gambar 4.22</b>	Perbandingan Kecepatan Motor <i>Brushless</i> DC Pengereman 50%.....	47

## DAFTAR TABEL

<b>Tabel 2.1</b>	Spesifikasi Motor <i>Brushless</i> DC Tipe <i>Outrunner</i> RCTimer A2830-13 850KV .....	6
<b>Tabel 2.2</b>	Deskripsi Arduino MEGA 2560.....	12
<b>Tabel 2.3</b>	Pengaruh Nilai $K_p$ , $K_i$ , $K_d$ Pada Respon Sistem.....	15
<b>Tabel 4.1</b>	Pengujian Sensor <i>Rotary Encoder</i> .....	35
<b>Tabel 4.2</b>	Pengujian Kecepatan Motor (Tanpa Pengereman).....	37
<b>Tabel 4.3</b>	Pengujian Kecepatan Motor (Dengan Setengah Pengereman).....	39
<b>Tabel 4.4</b>	Pengujian Kecepatan Motor (Dengan Pengereman Penuh) .....	41
<b>Tabel 4.5</b>	Data Kecepatan Tanpa Kontroler PID .....	44
<b>Tabel 4.6</b>	<i>History</i> Perubahan Nilai $K_p$ , $T_i$ , $T_d$ Pada Setiap Pengereman .....	45
<b>Tabel 4.7</b>	<i>History</i> Perubahan Nilai $K_p$ , $T_i$ , $T_d$ Pada Setiap Pengereman .....	45
<b>Tabel 4.8</b>	Perubahan Nilai $K_p$ , $T_i$ , $T_d$ Pada Setiap Pengereman ....	46
<b>Tabel 4.9</b>	Data Kecepatan Dengan Kontroler PID .....	46
<b>Tabel 4.10</b>	Hasil Kecepatan Motor <i>Brushless</i> DC Dengan Kontroler PID .....	48

*-- Halaman ini sengaja dikosongkan --*



*-- Halaman ini sengaja dikosongkan --*

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Motor *brushless* DC merupakan salah satu dari beberapa jenis motor DC. Jika ingin mengaplikasikan motor *brushless* DC pada proses pembelajaran, maka diperlukan alat yang berfungsi untuk mengetahui dan mengontrol kecepatan dari motor *brushless* DC tersebut. Selama ini pengontrolan mengenai motor *brushless* DC hanya sebatas kontrol kecepatan putaran motor yang dilakukan dengan berbagai metode kontrol. Namun masih belum ada alat atau modul kontrol kecepatan motor *brushless* DC yang khusus diperuntukkan untuk kegiatan praktikum.

Modul yang digunakan pada proses pembelajaran harus sederhana dan praktis, sehingga memudahkan pengguna untuk menggunakan modul ini. Selain sederhana dan praktis, juga harus akurat. Artinya, data yang dihasilkan oleh modul ini harus sesuai dengan perhitungan yang ada. Namun, modul khusus untuk kontrol kecepatan motor *brushless* DC tidak mudah untuk dibuat. Sementara itu masih banyak mahasiswa yang belum tahu bagaimana cara mengontrol motor *brushless* DC dengan benar.

Salah satu cara yang paling memungkinkan untuk melakukan kontrol kecepatan yaitu dengan menggunakan sensor *rotary encoder* yang dikopel dengan motor yang dikemas dalam bentuk modul pembelajaran dan dikontrol dengan menggunakan suatu metode kontrol. Data yang dihasilkan kemudian ditampilkan pada perangkat komputer atau laptop. Hal ini bertujuan mempermudah pengguna mengetahui hasil pengamatan yang telah mereka lakukan.

### **1.2 Permasalahan**

Permasalahan yang diangkat pada tugas akhir ini adalah ;

1. Masih banyak mahasiswa yang belum tahu bagaimana mengontrol kecepatan motor *brushless* DC dengan benar.
2. Belum adanya modul pembelajaran praktis guna mempermudah proses pembelajaran mengenai kontrol kecepatan motor *brushless* DC.

### **1.3 Batasan Masalah**

Pembatasan pada tugas akhir ini meliputi :

1. Arah putaran motor *brushless* DC tidak diperhatikan.

2. Kontroler yang digunakan adalah kontroler PID dengan metode *trial and error*.

#### **1.4 Tujuan**

Tujuan tugas akhir ini adalah :

1. Mengontrol kecepatan dari motor *brushless* DC.
2. Membuat modul pembelajaran praktis yang mudah digunakan sebagai sarana pembelajaran motor *brushless* DC.

#### **1.5 Sistematika Penulisan**

Dari proses pembuatan alat tugas akhir ini dimulai dari studi literatur, menentukan gambar desain alat, membuat alat, membuat program, menguji alat secara keseluruhan, analisa data, serta dapat menyusun laporan akhir dengan sistematika penulisan yaitu pendahuluan, teori penunjang, perancangan alat, pengukuran dan analisa serta penutup.

### **BAB I PENDAHULUAN**

Berisi latar belakang pembuatan alat, permasalahan, pembatasan masalah, tujuan, metodologi, sistematika penulisan serta manfaat Tugas Akhir.

### **BAB II TEORI PENUNJANG**

Meliputi pembahasan dan teori-teori penunjang yang digunakan dalam mendukung terselesaikannya Tugas Akhir. Teori-teori yang dipakai dalam pembuatan proyek Tugas Akhir ini antara lain : motor *brushless* DC, sensor *rotary encoder*, pengereman magnetik, arduino MEGA 2560, *interfacing*, serta teori penunjang lainnya.

### **BAB III PERANCANGAN ALAT**

Membahas tentang tahap-tahap perancangan mekanik dan perancangan sistem kontrolernya.

### **BAB IV PENGUKURAN DAN ANALISA**

Membahas tentang pengukuran dan pengujian alat atau rangkaian yang digunakan pada tugas akhir ini.

### **BAB V PENUTUP**

Berisikan tentang kesimpulan atas hasil yang diperoleh serta saran-saran atas kekurangan dan kelemahan tugas akhir ini.

## **1.6 Relevansi**

Relevansi yang diterapkan dari tugas akhir ini adalah :

1. Memenuhi kurikulum dari kuliah Diploma 3 (tiga).
2. Memudahkan proses pembelajaran dalam kontrol kecepatan motor *brushless* DC.

***-- Halaman ini sengaja dikosongkan --***

## BAB II

### TEORI PENUNJANG

Sistem kendali atau sistem kontrol adalah suatu alat (kumpulan alat) untuk mengendalikan, memerintah, dan mengatur keadaan dari suatu sistem. Banyak contoh dalam bidang industri/instrumentasi dan dalam kehidupan kita sehari-hari di mana sistem ini dipakai, salah satunya adalah sistem kendali/kontrol kecepatan pada motor *brushless* DC.

Pada bab ini akan dibahas mengenai teori-teori yang berkaitan dengan peralatan yang akan dirancang. Teori yang mendukung penyelesaian tugas akhir ini diantaranya adalah mengenai : motor *brushless* dc, sensor *rotary encoder*, pengereman magnetik, mikrokontroler arduino MEGA 2560, jaringan *wireless* Ad Hoc, LabView, dan kontroler PID.

#### 2.1 Motor *Brushless* DC [1]

Salah satu jenis motor listrik yang paling banyak digunakan akhir-akhir ini adalah motor *brushless* DC dimana motor DC ini tidak menggunakan *brush* (sikat) untuk proses komutasi. Motor *brushless* DC sangat cocok untuk diaplikasikan pada produk yang menuntut reliabilitas dan efisiensi yang tinggi. Secara umum, dapat dikatakan bahwa motor *brushless* DC dapat menghasilkan torsi yang besar dan mempunyai *range* RPM yang tinggi. Motor *brushless* DC merupakan salah satu jenis motor sinkron dimana medan magnet yang dihasilkan oleh stator dan medan magnet yang dihasilkan oleh rotor berputar pada frekuensi yang sama. Untuk lebih jelasnya mengenai motor ini dapat dilihat pada Gambar 2.1 dan Tabel 2.1.

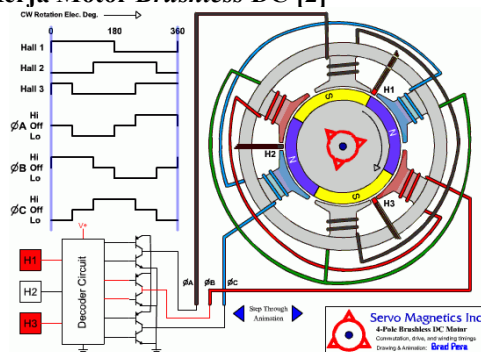


**Gambar 2.1** Salah Satu Contoh BLDC

**Tabel 3.1** Spesifikasi Motor *Brushless* DC Tipe *Outrunner* RCTimer A2830-13 850KV [2]

No.	Parameter	Nilai	
1.	Kecepatan Putar	12580 rpm	
2.	Berat Motor	52 gram	
3.	KV	850 rpm/Volt	
4.	Tegangan	Tegangan Minimal	7,40 Volt
		Tegangan Maksimal	14,80 Volt
5.	Input Arus	Arus Nominal	13 Ampere
		Arus Maksimal	25 Ampere
6.	Maximal Power	187 Watt	
7.	Input Baterai	Lithium Polimer 2S-4S	

### 2.1.1 Cara Kerja Motor *Brushless* DC [2]

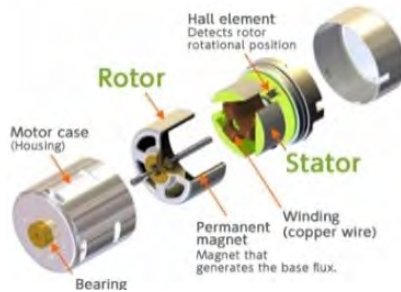


**Gambar 2.2** Skema Kerja Motor *Brushless* DC

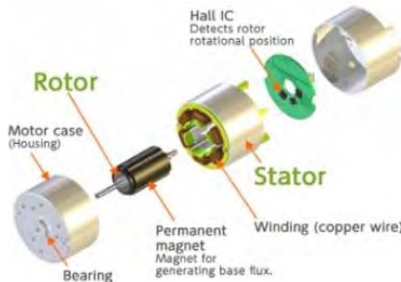
Cara kerja pada motor *brushless* DC cukup sederhana, yaitu magnet yang berada pada poros motor akan tertarik dan terdorong oleh gaya elektromagnetik yang diatur oleh *Electronic Speed Kontroller* (ESC). Hal ini yang membedakan motor *bushless* DC dengan motor DC dimana motor DC menggunakan sikat mekanis yang berada pada komutator untuk mengatur waktu dan memberikan medan magnet pada lilitan. Motor *brushless* DC ini juga berbeda dengan motor AC yang pada umumnya menggunakan siklus tenaga sendiri untuk mengatur waktu dan memberi daya pada lilitan. Motor *brushless* DC dapat memberikan rasio daya dan beban yang lebih tinggi secara signifikan

dan memberikan efisiensi yang lebih baik dibandingkan motor tanpa sikat tradisional.

### 2.1.2 Konstruksi Motor *Brushless* DC [1]



**Gambar 2.3** *Outer Construction*



**Gambar 2.4** *Inner Construction*

Setiap motor *brushless* DC memiliki dua bagian utama, rotor (bagian berputar) dan stator (bagian stasioner). Bagian penting lainnya dari motor adalah gulungan stator dan magnet rotor.

a. Rotor

Rotor adalah bagian pada motor yang berputar karena adanya gaya elektromagnetik dari stator, dimana pada motor *brushless* DC bagian rotornya berbeda dengan rotor pada motor DC konvensional yang hanya tersusun dari satu buah elektromagnet yang berada diantara *brushes* (sikot) yang terhubung pada dua buah motor hingga delapan pasang kutub magnet permanen berbentuk persegi panjang yang saling direkatkan menggunakan semacam “*epoxy*” dan tidak ada *brushes*-nya.



b. Stator

Stator adalah bagian pada motor yang diam/statis dimana fungsinya adalah sebagai medan putar motor untuk memberikan gaya elektromagnetik pada rotor sehingga motor dapat berputar. Pada motor *brushless* DC statornya terdiri dari 12 belitan (elektromagnet) yang bekerja secara elektromagnetik dimana statornya terhubung dengan tiga buah kabel untuk disambungkan pada rangkaian kontrol sedangkan pada motor DC konvensional statornya terdiri dari dua buah kutub magnet permanen. Belitan stator pada motor *brushless* DC terdiri dari dua jenis, yaitu belitan stator jenis *trapezoidal* dan jenis *sinusoidal*. Yang menjadi dasar perbedaan kedua jenis belitan stator tersebut terletak pada hubungan antara koil dan belitan stator yang bertujuan untuk memberikan EMF (*Electro Motive Force*) balik yang berbeda. EMF balik sendiri adalah tegangan balik yang dihasilkan oleh belitan motor *brushless* DC ketika motor tersebut berputar yang memiliki polaritas tegangan berlawanan arahnya dengan tegangan sumber yang dibangkitkan, sehingga besarnya EMF balik dapat dihitung dengan persamaan :

$$\text{EMF balik} = B \cdot N \cdot l \cdot r \cdot \omega \dots\dots\dots (2.1)$$

di mana :

B : kerapatan medan magnet yang dihasilkan rotor (Tesla)

N : banyaknya lilitan pada belitan stator per fasa

l : panjangnya batang rotor (m)

R : jari-jari dalam motor (m)

$\Omega$  : kecepatan sudut putaran motor (rad) (dimana  $\omega = 2\pi f$ )

Perubahan besarnya EMF balik ini mempengaruhi torsi motor *brushless* DC, sebagaimana rumus torsi pada motor *brushless* DC menurut persamaan diatas bahwa besarnya torsi yang dihasilkan motor *brushless* DC dapat dihitung dengan :

$$T = K_{rms} \cdot \Phi \cdot I \text{ (Nm)} \dots\dots\dots (2.2)$$

di mana :

$K_{rms}$  : tegangan rata-rata konstan (Volt)

$\Phi$  : besarnya fluks magnet (Tesla)

I : besarnya arus (Ampere)

c. *Axle*

*Axle* atau sumbu adalah batang yang berfungsi sebagai sumbu putar motor, terpusat pada rotor dan dirangkai bersama rotor.

d. *Sensor Hall*

Tidak seperti motor DC, *brushed* komutasi dari motor *brushless* DC diatur secara elektronik agar motor dapat berputar, stator harus di-*energize* secara berurutan dan teratur. *Sensor hall* inilah yang berperan dalam mendeteksi pada bagian rotor mana yang ter-*energize* oleh fluks magnet sehingga proses komutasi yang berbeda (enam langkah komutasi) dapat dilakukan oleh stator dengan tepat karena sensor *hall* ini dipasang menempel pada stator. *Hall sensor* ini ditempatkan setiap  $120^\circ$  pada jarak antar kutub stator hal ini bertujuan agar deteksi terhadap *vector* fluks stator yang dihasilkan akurat setiap perpindahan komutasi, arus yang mengalir tetap terjaga konstan pada setiap fasa. Prinsip kerja *hall sensor* sendiri membutuhkan arus yang mengalir terus jika ingin digunakan sebagai pendeteksi fluks magnet.

### 2.1.3 ESC (*Electronic Speed Controller*) [1]



**Gambar 2.5** ESC

ESC merupakan rangkaian elektronik yang berfungsi sebagai pengatur kecepatan putaran motor pada motor RC, cara kerjanya yaitu dengan cara menterjemahkan sinyal yang diterima *receiver* dari *transmitter*. Di pasaran terdapat berbagai *merk* ESC dengan kekuatan arus (*current rating*) dan kekuatan Voltase (*voltage rating*) serta *feature* yang ditawarkan.

Penjelasan yang cepat mengenai ESC adalah ESC harus secara akurat menghubungkan dan memutuskan koneksi antara 3 masukan *input* dan 3 belitan pada stator agar rotor dapat berputar.

Untuk menentukan ESC yang akan kita gunakan sangatlah penting untuk mengetahui kekuatan (*peak current*) dari motor. Pilihlah ESC yang kekuatannya melebihi kekuatan motor. Misalnya, dari data kita

dapatkan kekuatan motor adalah 12A (sesuai dengan *datasheet* motor) pada saat *throttle* terbuka penuh. Sebaiknya ESC yang akan kita gunakan adalah ESC yang berkekuatan 18A atau 20A. Jika kita paksakan menggunakan ESC 10A kemungkinan pada saat *throttle* dibuka penuh, ESC akan panas bahkan terbakar.

## 2.2 Sensor Rotary Encoder [1]



**Gambar 2.6** Rotary Encoder

*Rotary encoder* adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi angular pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitoring akan diubah ke dalam bentuk data digital oleh *rotary encoder* berupa lebar pulsa kemudian akan dihubungkan ke mikrokontroler.

Konstruksi *rotary encoder* berupa piringan tipis yang biasanya di kopel dengan poros yang berputar. Piringan tipis tersebut terdapat lubang di sepanjang pinggir lingkarannya. Di bagian sisi-sisi piringan terdapat sebuah led dan *phototransistor* di bagian bersebrangan. Fungsi dari lubang-lubang yang berada di sepanjang pinggir lingkaran tersebut akan menghantarkan cahaya led ke *phototransistor*, sebaliknya jika cahaya led tidak menembus lubang piringan maka cahaya akan tertahan. Piringan tersebut akan berputar sesuai dengan kecepatan putaran motor sehingga *phototransistor* akan saturasi ketika cahaya led menembus lubang-lubangnya.

Pada saat saturasi *phototransistor* akan menghasilkan pulsa dengan *range* +0,5 V s/d +5 V. Semakin banyak lubang yang berada pada piringan tentu saja semakin banyak pulsa yang dihasilkan selama satu putaran, hal tersebut berbanding lurus dengan tingkat akurasi yang dihasilkan oleh *rotary encoder*.

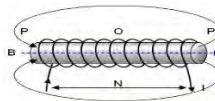
## 2.3 Pengereman Magnetik [2]

Rem adalah suatu alat yang digunakan untuk melakukan aksi deselerasi yang akan menurunkan kecepatan dalam selang waktu yang ditentukan. Tipe rem yang umumnya digunakan adalah rem yang

menggunakan gaya gesek untuk memberikan gaya lawan terhadap gaya gerak. Ada juga tipe rem lain yang tidak memanfaatkan gesekan dua permukaan untuk menghasilkan gaya lawan terhadap gaya penyebab gerak, yaitu rem yang menggunakan gaya magnet untuk menimbulkan gaya lawan. Rem ini disebut Rem Arus Eddy (Rem Magnetik).

Prinsip dasar rem magnetik ini menggunakan hukum Faraday dan hukum Lenz yang sudah terkenal di dunia elektromagnetik. Kedua hukum ini menimbulkan arus eddy yang melingkar dan menginduksi medan magnet yang melawan medan magnet penyebabnya. Hukum-hukum ini berlaku bila ada permukaan yang memotong medan magnet, dengan artian gaya lawan hanya dihasilkan apabila permukaan tersebut memiliki kecepatan. Semakin tinggi kecepatan maka gaya lawan yang dihasilkan juga semakin besar. Namun semakin rendah kecepatan maka gaya lawan akan semakin kecil.

Komponen dasar rem magnetik adalah sumber magnet yang diposisikan diam dan konduktor diamagnetik (tidak memiliki sifat magnet tetapi memiliki sifat penghantar listrik) yang bergerak dan tergabung dengan sumbu (*shaft*). Penggunaan elektromagnet dapat memanfaatkan kawat yang dililitkan pada inti besi sehingga menghasilkan kutub utara dan kutub selatan magnet. Kumparan ini sering disebut *solenoida*. Garis medan magnet didalam *solenoida* ditunjukkan dalam Gambar 2.7.



**Gambar 2.7** Garis Medan Magnet Di Dalam *Solenoida*

Dua buah *solenoida* yang berbeda kutub dipasangkan sehingga membentuk medan magnet. Konduktor disisipkan diantara kedua *solenoida* tersebut. Karena konduktor terhubung dengan shaft yang bergerak dengan kata lain memiliki kecepatan sehingga konduktor ini akan memotong medan magnet yang dihasilkan oleh kedua *solenoida* tersebut. Dapat disimpulkan bahwa, gaya gerak listrik (GGL) dapat ditimbulkan oleh adanya perubahan fluks magnet tiap detik. Pernyataan matematisnya adalah :

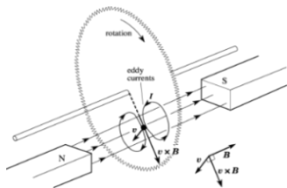
$$E = - \frac{d\Phi}{dt} \dots\dots\dots (2.3)$$

di mana :

E : GGL

$\Phi$  : fluks magnet

Tanda negatif menunjukkan arah E yang selalu melawan penyebabnya.



**Gambar 2.8** Gaya Pengereman Yang Dihasilkan Oleh Arus Melingkar Eddy

**2.4 Mikrokontroler Arduino MEGA 2560 [2]**

Arduino MEGA 2560 adalah *board* arduino yang merupakan perbaikan dari *board* arduino MEGA sebelumnya. Arduino MEGA awalnya memakai *chip* ATmega1280 dan kemudian diganti dengan *chip* ATmega2560, oleh karena itu namanya diganti menjadi arduino MEGA 2560. Tabel 2.2 merupakan spesifikasi arduino MEGA 2560.

**Tabel 2.2**Deskripsi Arduino MEGA 2560

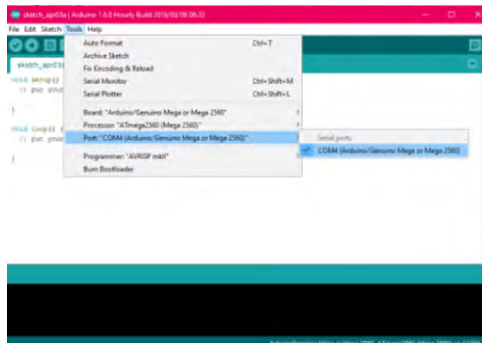
Mikrokontroler	RISC ATmega 2560
<i>Operating Voltage</i>	5 V
<i>Input Voltage (recommended)</i>	7 – 12 V
<i>Input Voltage (limit)</i>	6 – 20 V
<i>Digital I/O Pins</i>	54 (15 diantaranya <i>input</i> PWM)
<i>Analog Input Pins</i>	16
<i>DC Current per I/O Pin</i>	40 mA
<i>DC Current for 3,3V Pin</i>	50 mA
<i>Flash Memory</i>	256 KB (8 KB sebagai <i>bootloader</i> )
SRAM	8 KB
EEPROM	4 KB
<i>Clock Speed</i>	16 MHz
<i>USB Host Chip</i>	MAX3421E
<i>Length</i>	101,52 mm
<i>Width</i>	53,3 mm
<i>Weight</i>	37 g



**Gambar 2.9** Arduino MEGA 2560

Untuk memprogram arduino juga harus dilakukan beberapa tahapan sebagai berikut :

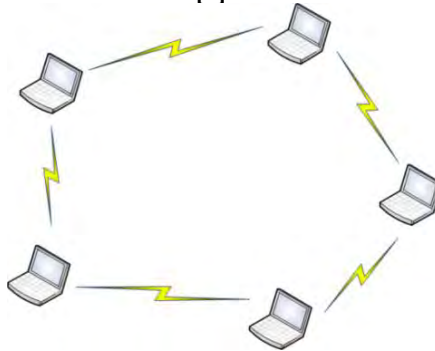
1. **Setting Board Arduino.** Dalam pemrograman *software* arduino harus di *setting* terlebih dahulu *board* arduino agar penggunaan arduino cocok. Dalam purwarupa kali ini arduino menggunakan arduino MEGA 2560. Untuk *setting board* arduino bisa masuk ke *tools – board* – setelah itu pilihlah *board* arduino yang sesuai.
2. **Setting Serial.** *Serial* ini merupakan kabel arduino yang dihubungkan kepada komputer atau laptop. *Serial* ini mempunyai dua fungsi yang bisa digunakan. Pertama *serial port* digunakan untuk mendownload program dari arduino yang kedua *serial* digunakan sebagai komunikasi *serial* pada arduino dengan komputer. *Setting serial* bisa masuk *tools – serial* - lali pilih COM yang sesuai dengan arduino yang terpasang. Untuk lebih jelasnya dapat dilihat pada Gambar 2.10.



**Gambar 2.10** Setting Serial Port

3. Apabila *program* tidak dapat di *download* karena *serial port*, maka cek terlebih dahulu *serial* yang benar pada *device manager*. Lalu dalam *software* arduino untuk memilih *serial port*nya samakan dengan *serial port* untuk arduino dalam *device manager* tersebut. Untuk masuk ke *device manager* dapat masuk start windows – lalu ketika *device manager* klik dua kali dan masuk ke dan COM.

## 2.5 Jaringan *Wireless Ad Hoc* [3]



**Gambar 2.11** Jaringan *Wireless Ad Hoc*

Jaringan *wireless Ad Hoc* adalah kumpulan *node* (atau *router*) *wireless mobile* yang secara dinamis keberadaannya tanpa menggunakan jaringan infrastruktur yang ada atau administrasi yang terpusat. Jaringan *wireless Ad Hoc* dapat juga dikatakan sebagai desentralisasi jaringan *wireless*.

Istilah *jaringan wireless Ad Hoc* sebelumnya dikenal dengan jaringan *packet radio* (PRNET) sekitar tahun 1970 dimana ini disponsori oleh DARPA dan proyek ALOHAnet.

Pada jaringan *Ad Hoc*, *router* dapat dengan bebas melakukan organisasi jaringan yang berakibat topologi akan berubah dengan cepat dan sulit untuk diprediksi. Dengan fitur ini, jaringan *Ad Hoc* mengalami beberapa tantangan antara lain :

- a. Multihop
- b. *Mobility*
- c. Kombinasi jaringan yang besar dengan berbagai peralatan yang berbeda
- d. *Bandwidth*

- e. Keterbatasan konsumsi baterai

Terlepas permasalahan diatas, jaringan Ad Hoc juga memerlukan protokol *routing* karena setiap *node* memerlukan pertukaran data contohnya seperti terlihat pada Gambar 2.11.

## 2.6 LabView [4]

LabView adalah sebuah *software* pemrograman yang diproduksi oleh *National instruments* dengan konsep yang berbeda. Seperti bahasa pemrograman lainnya yaitu C++, MATLAB atau *Visual basic*, LabView juga mempunyai fungsi dan peranan yang sama, perbedaannya bahwa LabView menggunakan bahasa pemrograman berbasis grafis atau blok diagram sementara bahasa pemrograman lainnya menggunakan basis text. Program LabView dikenal dengan sebutan Vi atau *Virtual instruments* karena penampilan dan operasinya dapat meniru sebuah *instrument*. Pada LabView, *user* pertama-tama membuat *user interface* atau *front panel* dengan menggunakan kontrol dan indikator, yang dimaksud dengan kontrol adalah *knobs*, *push buttons*, *dials* dan peralatan *input* lainnya sedangkan yang dimaksud dengan indikator adalah *graphs*, LEDs dan peralatan *display* lainnya. Setelah menyusun *user interface*, lalu *user* menyusun blok diagram yang berisi kode-kode VIs untuk mengontrol *front panel*.

## 2.7 Kontroler PID [5]

PID dari singkatan (*Propotional-Integral-Derivative Controller*) merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tersebut. Komponen control PID ini terdiri dari tiga jenis yaitu *Propotional*, *integral*, dan *derivative*. Ketiganya dapat dipakai bersamaan maupun sendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu *plant*.

Efek dari setiap pengendali dalam sistem *close loop* dapat dilihat pada Tabel 2.3 berikut.

**Tabel 2.3** Pengaruh Nilai  $K_p$ ,  $K_i$ ,  $K_d$  Pada Respon Sistem

Respon <i>Close Loop</i>	<i>Rise Time</i>	<i>Overshoot</i>	<i>Settling Time</i>	<i>Error Steady State</i>
$K_p$	Turun	Naik	Perubahan Kecil	Turun
$K_i$	Turun	Naik	Naik	Hilang
$K_d$	Perubahan Kecil	Turun	Turun	Perubahan Kecil



Blok diagram dari kontroler PID dengan sistem *close loop* digambarkan pada Gambar 2.12 dibawah ini.



**Gambar 2.12** Blok Diagram Kontroler PID Sistem *Close Loop*

Kontroler PID dibagi menjadi dua, yakni PID *Continuous* dan PID *Discrete*. Berikut akan dibahas tentang kontroler PID *Continuous*.

### 2.7.1 Kontroler PID *Continuous* [5]

Pada bagian ini akan dibahas tentang kontroler P, kontroler I, dan kontroler D. pada kontroler PID *Continuous* ini respon motor selalu memiliki nilai pada setiap waktu, karena sinyal yang terbentuk selalu membuat sinyal kontinyu. Kontroler P digunakan sebagai perintah pada motor DC untuk cepat mencapai *rise time*. Pada kontroler P ini bisa mempengaruhi sistem yaitu menambah atau mengurangi kestabilan, dapat memperbaiki respon transien khususnya adalah *rise time* dan *time settling*. Kontroler I digunakan untuk mengurangi error *steady*, respon motor lebih lambat karena dapat mencapai *steady state*, dan juga pada kontroler I dapat menambah ketidakstabilan karena menambah orde pada sistem. Sedangkan kontroler D hanya digunakan apabila terjadi sinyal error. Karena kontroler D ini bekerja apabila ada error, memberikan efek redaman pada sistem yang berisolasi sehingga bisa memperbesar pemberian nilai  $K_p$ , serta juga dapat untuk memperbaiki respon transien karena bekerja pada saat error. Oleh karena itu, kontroler D ini tidak dapat berdiri sendiri dalam penggunaanya.

Apabila menggunakan kontroler P saja, maka respon motor akan cepat untuk mencapai *rise time* akan tetapi akan lama untuk mencapai *steady state* karena kontroler P ini member pengaruh langsung (sebanding) pada error. Semakin besar error, semakin besar sinyal kendali yang dihasilkan kontroler. Berikut adalah perumusan dari kontroler P :

$$P = K_p = \frac{e}{K_p \cdot \tau} \dots \dots \dots (2.4)$$

Apabila menggunakan PI, maka respon motor akan cepat untuk mencapai *rise time* dan juga akan mencapai *steady state* karena kontroler

I ini bersifat untuk mengurangi eror *steady state* walaupun dengan waktu yang lama untuk mencapai *steady state*-nya. Berikut adalah perumusan kontroler PI :

$$PI = K_p \left(1 + \frac{1}{\tau_i s}\right) \dots \dots \dots (2.5)$$

Apabila menggunakan PD, maka respon motor akan cepat mencapai *rise time* dan langsung bisa mencapai *steady state* akan tetapi respon tidak stabil karena hanya dapat dipergunakan kalau terdapat eror saja. Berikut adalah perumusan kontroler PD (orde 2) :

$$P(t) = K_p \left( e(k) + \tau_d \frac{de(t)}{dt} \right) \dots \dots \dots (2.6)$$

Apabila menggunakan kontroler PID maka respon motor akan cepat untuk mencapai *rise time* dan juga cepat mencapai *steady state* karena kontroler mengurangi eror *steady state*, sedangkan kontroler D ini bisa bekerja karena terdapat eror dari kontroler P dan kontroler I sehingga motor dapat mencapai kondisi yang stabil. Perumusan dan kontroler PID *Continuous* ini adalah sebagai berikut :

$$U(s) = K_p \left\{ 1 + \frac{1}{\tau_i s} + \tau_d s \right\} \dots \dots \dots (2.7)$$

***-- Halaman ini sengaja dikosongkan --***

## **BAB III**

### **PERANCANGAN ALAT**

Dalam bab ini akan dibahas mengenai perancangan alat yang meliputi perencanaan perangkat keras (*hardware*) dan perangkat lunak (*software*). Hal tersebut guna mewujudkan tugas akhir yang berjudul “Pembuatan Modul Kontrol Kecepatan Motor *Brushless* DC Dengan Mikrokontroler”. Perancangan alat akan dibahas perbagian yang disertai dengan gambar skematik.

Untuk memudahkan dalam pembahasan bab ini akan dibagi menjadi 3 yaitu:

1. Blok fungsional sistem
2. *Setting* jaringan *wireless* Ad Hoc
3. Perancangan perangkat keras yang terdiri dari perancangan mekanik dan elektrik, yaitu:
  - a. Perancangan mekanik meliputi :
    1. Desain Alat Secara Keseluruhan
    2. Rem Magnetik
  - b. Perancangan elektrik meliputi :
    1. Rangkaian Sensor *Rotary Encoder*
    2. *Driver* Rem Magnetik
4. Perancangan perangkat lunak yang terdiri dari :
  - a. Perancangan perangkat lunak dengan *software* Arduino MEGA 2560 yang menggunakan bahasa C sebagai bahasa pemrogramannya.
  - b. Perancangan perangkat lunak LabView

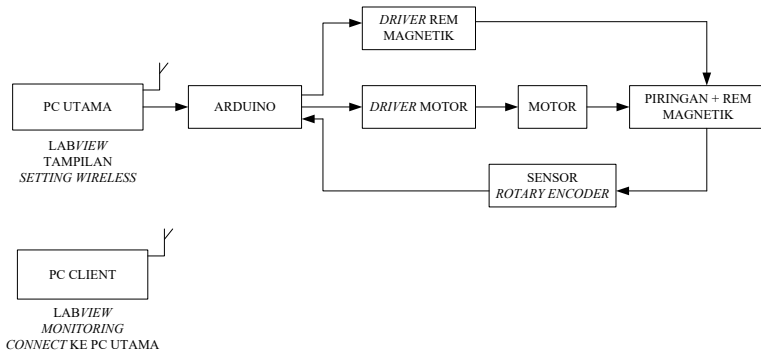
#### **3.1 Blok Fungsional Sistem**

Perancangan sistem dalam pembuatan alat ini secara garis besar disertai urutan dan cara kerja alat ini di ilustrasikan pada Gambar 3.1 dan Gambar 3.2.

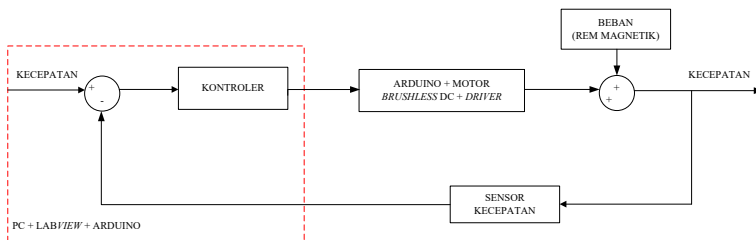
Dari Gambar 3.1 dapat dilihat bahwa sistem tersebut terdiri dari beberapa blok fungsional yaitu;

1. Arduino MEGA 2560, merupakan mikrokontroler yang berfungsi sebagai *interface* dari perangkat elektronik dan dapat menyimpan program didalamnya.
2. *Driver*, berfungsi sebagai penggerak motor, rem dan sensor.
3. Motor *brushless* DC, digunakan sebagai objek yang akan dikontrol kecepatannya.
4. Rem magnetik, digunakan sebagai media pengereman dan juga sebagai beban bagi motor *brushless* DC.

5. Sensor kecepatan (*rotary encoder*), digunakan untuk mengetahui sampai seberapa kecepatan motor yang terjadi.
6. LabView, merupakan perangkat lunak yang digunakan sebagai media untuk memberi masukan atau interupsi ke kontroler.
7. Tampilan (*Display*), untuk tampilan digunakan PC, PC ini digunakan untuk menampilkan kecepatan motor *brushless* DC.



**Gambar 3.1** Blok Diagram Perancangan



**Gambar 3.2** Blok Diagram Sistem *Close Loop* PID

### 3.2 Setting Jaringan Wireless Ad Hoc

Ad Hoc merupakan salah satu *mode* jaringan dalam WLAN (*Wireless Local Area Network*). *Mode* ini memungkinkan dua atau lebih *device* (komputer atau *router*) untuk saling berkomunikasi satu sama lain secara langsung (dikenal dengan istilah *peer to peer*) tanpa melalui *Central Wireless Router* atau *Acces Point* (AP).

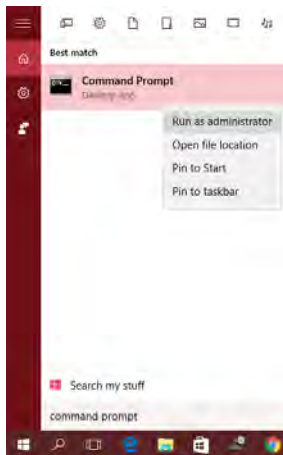
Ad Hoc didasari pada spesifikasi IEEE 802.11. Saat ini ada empat variasi dari 802.11, yaitu: 802.11a, 802.11b, 802.11g dan 802.11n yang mempunyai data rate up to 300Mbps (downlink) and 150Mbps (uplink).

Pada jaringan *wireless* Ad Hoc, *transmitter* menggunakan sinyal radio untuk berkomunikasi dengan menggunakan frekuensi yang sama. Secara umum, setiap *node* hanya akan menjadi *transmitter* (TX) atau *receiver* (RX) pada satu waktu artinya tidak bisa menjadi TX dan RX dalam satu waktu yang sama. Selain itu, komunikasi antara *mobile node* dibatasi oleh jangkauan transmisi.

Tidak seperti jaringan kabel, *packet delay* yang terjadi pada jaringan *wireless* Ad Hoc terjadi tidak hanya karena beban trafik tetapi juga beban trafik pada *node* yang didekatnya, ini dikenal dengan *traffic interference*.

Berikut adalah tahapan untuk membuat *wireless* Ad Hoc pada Windows 10 :

- a. Pertama kali yang dilakukan adalah memeriksa apakah komputer kita mendukung untuk membuat *wireless* Ad Hoc. Caranya dengan membuka *Start Screen* lalu ketikkan "CMD", pilih "*Command Prompt*" lalu klik kanan dan klik *Run As Administrator*.



**Gambar 3.3** *Start Screen* Windows 10

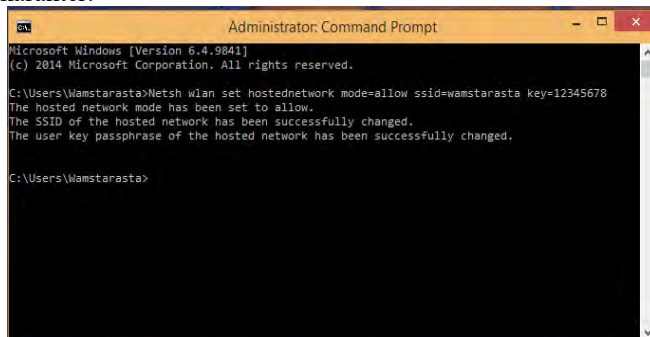
- b. Kemudian akan muncul jendela *Command Prompt*. Lalu ketikkan : "*netsh wlan show drivers*". Kemudian tekan tombol *enter* dan periksa baris *Hosted network supported* , jika "*Yes*" maka komputer sudah mendukung untuk membuat

*wireless* Ad Hoc , namun jika "No" maka perangkat keras komputer harus di-upgrade.



**Gambar 3.4** *Command Prompt*

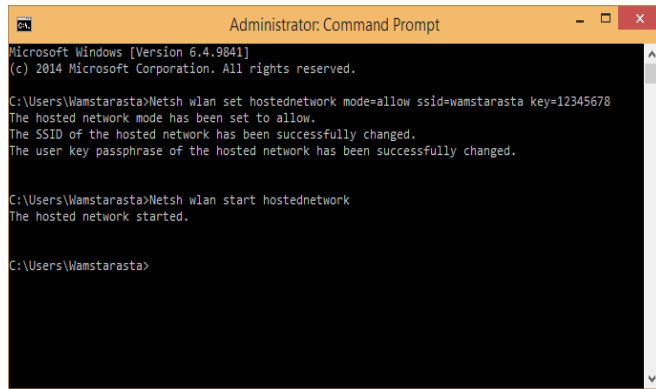
- c. Langkah selanjutnya, masih di *Command Prompt* dengan *mode Run as Administrator*, ketikkan "*netsh wlan set hostednetwork mode=allow ssid=NAMA JARINGAN key=KATA SANDI*". Kemudian tekan tombol *enter*. Untuk ssid adalah nama *wireless* Ad Hoc, sedangkan *key* adalah kata sandi dari *wireless network*. Kata sandi minimal memiliki 8 karakter.



**Gambar 3.5** *SSID Dan Key*

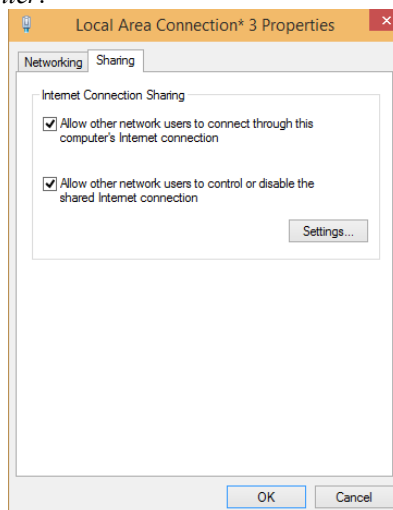
- d. Langkah selanjutnya adalah memulai *wireless* Ad Hoc yang dibuat yaitu dengan mengetikkan "*netsh wlan start*

*hostednetwork*" pada *commad prompt* dan menekan tombol *enter*.



**Gambar 3.6** Memulai *Wireless Ad Hoc*

- e. Langkah terakhir adalah mengaktifkan *Internet Connection Sharing*. Caranya adalah dengan mengklik kanan pada *icon* jaringan pada *taksbar*, kemudian pilih *Open Network and Sharing Center*, kemudian akan muncul jendela *Network and Sharing Center*.



**Gambar 3.7** *Local Area Connection*



Lihat pada bagian kiri, pilih *Change Adapter Setting*, selanjutnya klik kanan pada WiFi dan pilih *properties*. Pilih *tab Sharing* kemudian isi centang pada "*Allow other network users to connect through this computer's internet connection*", kemudian klik OK. Ulangi untuk mencentang "*Allow other network users to connect through this computer's internet connection* " kemudian pilih "*Local Area Connection \**" dan Klik OK.

### 3.3 Perancangan Perangkat Keras (*Hardware*)

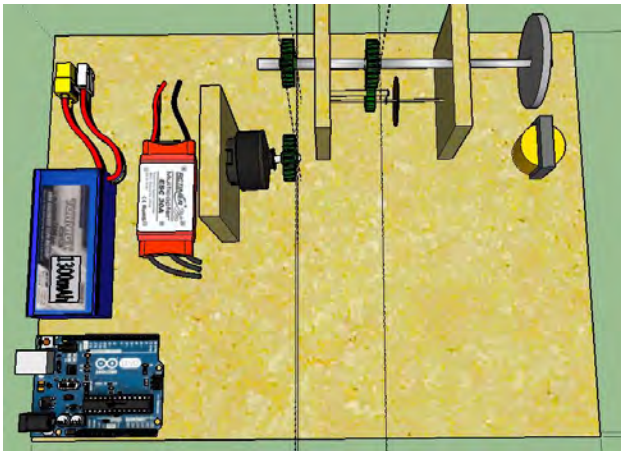
Pada perancangan perangkat keras ini, prosesnya dibagi menjadi 2 bagian, yaitu perancangan mekanik dan elektrik. Masing-masing perancangan tersebut selanjutnya akan dibahas lebih mendalam pada sub bab berikutnya.

#### 3.3.1 Perancangan Mekanik

Perancangan mekanik ini terdiri dari desain alat secara keseluruhan, motor *brushless* DC, rem magnetik, *rotary encoder*, dan *driver* motor *brushless* DC.

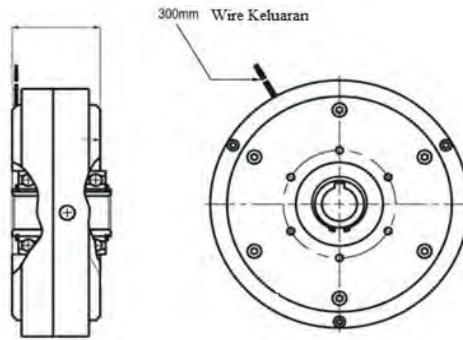
##### 3.3.1.1 Desain Alat Secara Keseluruhan

Gambar 3.8 akan menyatakan bagaimana desain modul kontrol kecepatan motor *brushless* DC secara keseluruhan.



**Gambar 3.8** Desain Modul Kontrol Kecepatan Motor *Brushless* DC Dengan Mikrokontroler

### 3.3.1.2 Rem Magnetik



**Gambar 3.9** Konfigurasi Rem Magnetik

Piringan aluminium dihubungkan dengan poros utama. Piringan ini diapit oleh dua penghasil medan magnet. Sisi penghasil medan magnet utara dan sisi penghasil medan magnet selatan. Saat poros utama berputar maka piringan aluminium akan ikut berputar. Pada saat kecepatan putar tinggi, medan magnet diberikan, sehingga akan menimbulkan arus eddy yang menimbulkan gaya lawan yang melawan arah putar piringan aluminium. Gaya lawan ini yang disebut sebagai beban, yang harus ditanggung motor *brushless* DC.

Pada sisi penghasil medan magnet utara digunakan 4 buah kumparan yang bila dialiri arus listrik akan menimbulkan medan magnet. Begitu juga pada sisi penghasil medan magnet selatan yang menggunakan 4 buah kumparan. Jumlah lilitan tiap kumparan dihitung berdasarkan rumus berikut ini :

$$\frac{V}{d} = \dots\dots\dots (3.1)$$

di mana :

N : jumlah lilitan tiap sisi

d : diameter kumparan (cm)

V : tegangan *input* (Volt)

Karena tiap sisi rem terdapat 4 kumparan maka setiap kumparan memiliki 283 lilitan. Kemudian semua kumparan dihubung seri. Sumber tenaga yang digunakan untuk rem magnetik ini sudah terkonfigurasi sama dengan sumber yang digunakan pada motor *brushless* DC. Besar

pengaturan tegangan yang masuk pada rangkaian rem magnet akan mempengaruhi banyak arus yang mengalir dan mempengaruhi besar medan magnet yang dihasilkan, yang berarti juga mempengaruhi gaya lawan yang dihasilkan.

### 3.3.2 Perancangan Elektrik

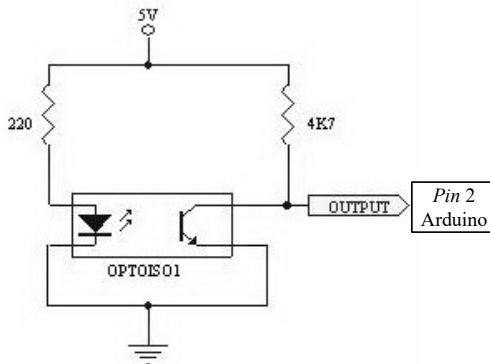
Perancangan elektronik ini meliputi desain layout PCB serta pengkabelan. Rangkaian elektrik pada *plant* ini meliputi rangkaian sensor *rotary encoder*, *driver* rem magnetik, dan arduino MEGA 2560.

#### 3.3.2.1 Rangkaian Sensor *Rotary Encoder*

*Rotary encoder* pada tugas akhir ini akan digunakan untuk menentukan banyaknya putaran poros tiap menit (RPM) yang kemudian akan menghasilkan gelombang kotak yang frekuensinya akan bertambah bila kecepatan putar poros bertambah. *Rotary encoder* ini diletakkan pada poros yang sudah dikopel dengan motor *brushless* DC.

*Rotary encoder* ini tersusun dari suatu piringan tipis yang memiliki lubang-lubang yang terdapat pada piringan tersebut. Setelah itu akan ditempatkan LED pada salah satu sisi piringan. Hal ini akan membuat cahaya masuk menuju piringan.

Kemudian disisi lain dari piringan ini diletakan *phototransistor* yang bertujuan untuk mendeteksi cahaya LED yang berseberangan. Piringan tipis ini yang nantinya akan dikopel dengan poros motor ataupun perangkat berputar lainnya yang ingin kita ketahui posisinya, hal ini akan membuat piringan berputar keika motor tersebut berputar. Apabila cahaya yang berasal dari LED mencapai *phototransistor*, maka *phototransistor* itu akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang pesergi.



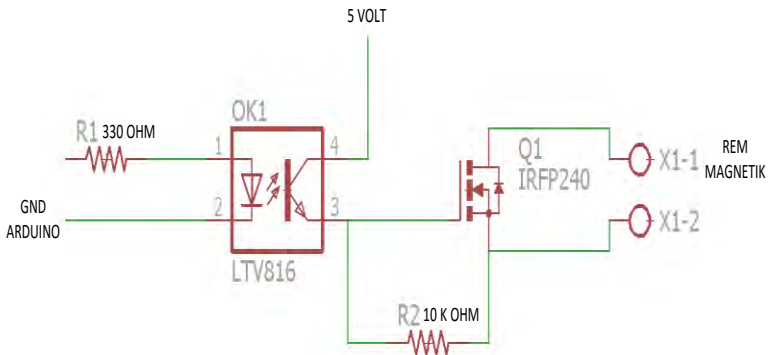
**Gambar 3.10** Rangkaian *Rotary Encoder*

Pada gambar diatas LED Inframerah kita gunakan untuk menembakkan cahaya sedangkan disisi kanan *light receive* dapat kita gunakan sensor cahaya seperti *photodiode* atau *phototransistor*.

Pada *transmitter* terdapat sebuah LED inframerah (IR LED) yang berfungsi untuk mengirimkan sinyal kepada *receiver*. Cahaya yang dipancarkan oleh LED inframerah tidak terlihat oleh mata telanjang. Sedangkan pada *receiver* terdapat *phototransistor* (transistor yang peka terhadap perubahan cahaya). Untuk memicu *optocoupler* maka digunakan piringan hitam yang diberi celah di ujungnya dan *dicouple* dengan poros *generator*. Prinsip kerjanya ketika generator berputar maka piringan akan ikut berputar, kemudian *optocoupler* menyensor ujung piringan. Ketika ada celah melewati *optocoupler*, maka cahaya dipancarkan *transmitter* menuju *receiver*, sehingga menghasilkan tegangan keluaran yang nilainya mendekati VCC, begitu juga sebaliknya, jika tidak ada benda diantara celah sensornya maka akan menghasilkan tegangan keluaran yang nilainya mendekati 0 Volt. Tujuannya adalah mendapatkan frekuensi putaran dari *generator*.

### 3.3.2.2 Driver Rem Magnetik

Rangkaian *driver* rem pada alat ini menggunakan IRFP-460 Power MOSFET. IRFP-460 adalah transistor efek medan semikonduktor logam-oksida / MOSFET (*Metal-Oxide Semiconductor Field-Effect Transistor*) tipe nMOS (*N-Channel*, kanal-N pada substrat-P) dengan *voltage rating*  $V_{(BR)DSS} = 500$  Volt, resistansi statis yang rendah ( $R_{DS(on)} < 0.27\Omega$ ) dan *drain current rating* sebesar  $I_D = 18,4A$ .



**Gambar 3.11** Driver Rem Magnetik

### 3.4 Perancangan Perangkat Lunak (*Software*)

Agar rem elektromagnetik dapat dikontrol dan hasil putaran dapat dilihat di LCD maka perlu dirancang di sebuah *software* yang mampu mengelola dan mengontrol data terhadap kinerja peralatan rem elektromagnetik. *Software* merupakan program berisi perintah-perintah yang dieksekusi oleh arduino MEGA 2560 sehingga sistem dapat bekerja sesuai dengan alur dan tujuan yang dirancang dan dapat ditampilkan pada komputer lain melalui jaringan *wireless* Ad Hoc.

#### 3.4.1 Pemrograman *Software* Arduino

Dalam perancangan program pada *software* arduino dengan fungsi terkait yang dibutuhkan diperlukan beberapa tahapan yang harus dilakukan terlebih dahulu. Tahapan tersebut adalah membuat algoritma dari alat yang sudah kita jalankan.

Pembuatan algoritma ini dilakukan setelah membuat *flowchart*, dari algoritma kita ini maka diharapkan alat yang akan dibuat ini dapat terlebih lebih sederhana. Setelah tahapan tersebut terselesaikan barulah kita memprogram fungsi terkait yang dikodingkan dalam bahasa C.

Berikut ini algoritma program utama dari rem elektromagnetik:

1. Modul dapat dioperasikan setelah sistem terpasang dengan benar seperti motor *brushless* DC beserta rem magnetik, *driver* motor *brushless* DC, *driver* rem magnetik, catu daya dan *potensiometer*.
2. Modul dapat bekerja jika rangkaian kontrol sudah dijalankan dan sudah terpasang.
3. *Potensiometer* digunakan untuk mengatur *range* yang akan di suplai ke *driver* rem elektromagnetik yang berupa tegangan dan akan di ubah menjadi digital melalui ADC Arduino.
4. *Driver* rem elektromagnetik akan mengatur tegangan yang masuk ke rem magnetik yang berfungsi untuk memperlambat putaran motor

Pada Tugas Akhir ini *software* yang digunakan adalah program arduino MEGA 2560 untuk membuat dan merencanakan program dalam bahasa C. Pemrograman *software* arduino dirancang dengan menggunakan *software* yang bernama Arduino IDE dengan menggunakan bahasa pemrograman C.

Arduino sangatlah berbeda sekali dengan mikrokontroler. Arduino merupakan sebuah *kit* mikrokontroler AVR yang dibuat dalam sebuah *board* (papan PCB). Dikembangkan di Italia sejak tahun 2005. Dalam 1 *board* sudah terdapat mikrokontroler lengkap dengan *pin/port* untuk

koneksi serta sudah dilengkapi dengan *downloader*. Dalam segi bahasa pemrograman, arduino memiliki bahasa pemrograman yang lebih mudah dan sederhana terutama bagi pemula.

Alasan bahasa pemrograman arduino lebih mudah dan sederhana adalah karena didalam arduino sudah terdapat beberapa *library* yang dapat digunakan untuk merancang pemrograman yang diinginkan. Pada Gambar 3.12 berikut menunjukkan *flowchart* yang digunakan pada Arduino MEGA 2560 pada tugas akhir ini.

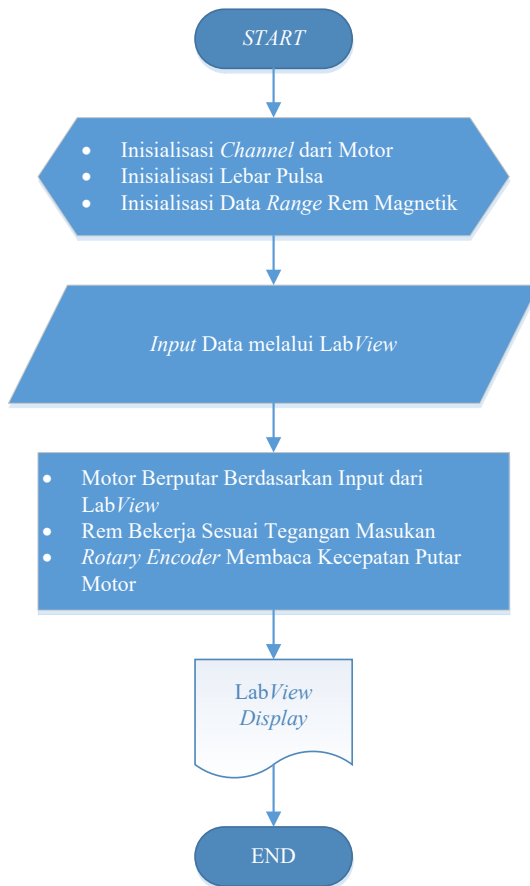


**Gambar 3.12** *Flowchart* Arduino MEGA 2560

### 3.4.2 Perancangan Perangkat Lunak LabView

Ketika awal memulai *software* LabView, tahapan yang dilakukan adalah dengan membuka *File Menu* kemudian klik *New VI*. *Untitled VI* akan mempunyai dua bagian yang disebut *Front Panel* dan *Block Diagram*. *Front Panel Window* merupakan *user interface window* pada saat VI dijalankan, sedangkan *Block Diagram Window* merupakan bagian yang akan melakukan kalkulasi dan mengeksekusi algoritma.

Gambar 3.13 menunjukkan *state diagram* yang difungsikan pada LabView untuk pengaturan kecepatan pada motor *brushless* DC.



**Gambar 3.13** Flowchart LabView

## BAB IV

### PENGUJIAN DAN ANALISA ALAT

Pada bab ini akan dibahas mengenai pengujian alat dan analisa data dari hasil rancangan alat yang telah dibuat. Pengujian alat ini ditujukan untuk memastikan agar peralatan dapat berfungsi dengan baik.

Pengujian dan analisa dari alat ini meliputi analisa program arduino, pengujian sensor *rotary encoder*, pengujian kecepatan putaran motor *brushless* DC tanpa menggunakan pengereman, pengujian kecepatan putaran motor *brushless* DC dengan menggunakan setengah pengereman, pengujian kecepatan putaran motor *brushless* DC dengan menggunakan pengereman penuh, serta pengujian motor *brushless* DC dengan kontroler PID LabView. Setelah melakukan beberapa pengujian tersebut, data yang diperoleh akan dianalisa untuk mengetahui proses kerja dari seluruh sistem alat yang dibuat.

#### 4.1 Analisa Program Arduino

Pada dasarnya program Arduino hanya berfungsi sebagai pengirim data dan penerima data dari komputer. Program Arduino yang dibuat kali ini memiliki memiliki 2 sistem utama yaitu inisialisasi dan program utama. Gambar 4.1 merupakan program inisialisasi yang dibuat.

Pada inisialisasi terdapat beberapa hal yang dilakukan yaitu menyebutkan jenis-jenis *library* yang dipakai, menyebutkan variabel-variabel yang dipakai, menyebutkan nilai *baudrate*, dan yang terakhir menyebutkan *pin-pin* yang dipakai pada Arduino.

Pada Gambar 4.1 bahwa dibutuhkan *library servo* dan *library TimerOne* untuk implementasi, *library servo* ini berfungsi untuk menjalankan motor sehingga motor hanya perlu dimasukkan nilai-nilai tertentu dan motor dapat langsung berputar sedangkan *library TimerOne* berfungsi sebagai *counter* untuk mengetahui berapa banyak pulsa yang dihasilkan dalam sekian waktu yang telah ditentukan..

Kemudian terdapat beberapa jenis variabel yang dipakai diantaranya *integer*, *float*, *unsigned long*, dan *unsigned int*. Dengan *baudrate* yang dipakai adalah bernilai 9600. Pada program utama Arduino akan terjadi program yang terus menerus dilakukan selama Arduino dalam kondisi menyala. Pada program utama terbagi menjadi 3 buah sistem, yang pertama merupakan program komunikasi data komputer (LabView) melalui USB *serial*, kemudian yang kedua merupakan program untuk membaca nilai kecepatan motor dalam satuan RPM, dan yang ketiga adalah merupakan program untuk *driver* pada rem magnetik.



```

volatile byte half_revolutions;
#include <Servo.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(22, 24, 26, 28, 30, 32);
#include <TimerOne.h>
unsigned int counter = 0;
int rotation, rotation1;
float xxx;
int sensorPin = A1;    // input analog potensiometer
unsigned long offset;

int sensorValue ;      // variable integer untuk menampung nilai potensiometer
unsigned long timeold;
Servo myservo;
unsigned int rpm;

//Min & max pulse
//int minPulse = 1000;
int maxPulse = 2000;
//char p[1023];
int speed = 0;

void setup()
{
  xxx = 0.0;
  Serial.begin(9600);
  arus = 0;
  lcd.begin(16, 2);
  pinMode(2, INPUT);
  Timer1.initialize(1000000);
  attachInterrupt(0, docount, RISING);
  Timer1.attachInterrupt( timerIsr );
  myservo.attach(8);    // pin 8 sebagai output ke ESC
  half_revolutions = 0;
  rpm = 0;
  timeold = 0;
  x = 0;
  myservo.write(x);
  for (i = 0; i < 1000; i++) {
    offset = offset + (analogRead(A0));
    delay(1);
  }
  offset = 512 - (offset / 1000);
  //rotation = 0;
}

```

**Gambar 4.1** Potongan Program Inisialisasi

Pada Gambar 4.2 ditunjukkan program untuk pengiriman dan penerimaan data antara komputer dan Arduino melalui USB *serial*. Kemudian pada Gambar 4.3 adalah program untuk membaca kecepatan motor *brushless* DC dalam satuan RPM.

```

if (Serial.available() > 0) {
  x = Serial.read();
  if (x < 0) {
    x = 0;
  }
  myservo.write(x);
}

```

**Gambar 4.2** Potongan Program Komunikasi Data

```

xxx = (rotation * 60 * 0.6668) - 224;
rotation1 = int(xxx);
if (rotation1 < 0) {
  rotation1 = 0;
  counter = counter / 4;
  rotation = counter;
}

```

**Gambar 4.3** Potongan Program Sensor Kecepatan

Pada program sensor kecepatan, program akan membaca berapa banyak jumlah pulsa yang dihasilkan dalam suatu waktu tertentu yang telah ditentukan. Kemudian jumlah pulsa ini akan dihitung dalam suatu rumus dan hasilnya akan dikonversi menjadi dalam satuan RPM. Setelah itu program akan menuliskan nilai kecepatan motor tersebut pada *front panel* LabView.

```

sensorValue = analogRead(sensorPin);
//delay(10);
sensorValue = map(sensorValue, 0, 1023, 0, 255);
analogWrite(9, sensorValue);

```

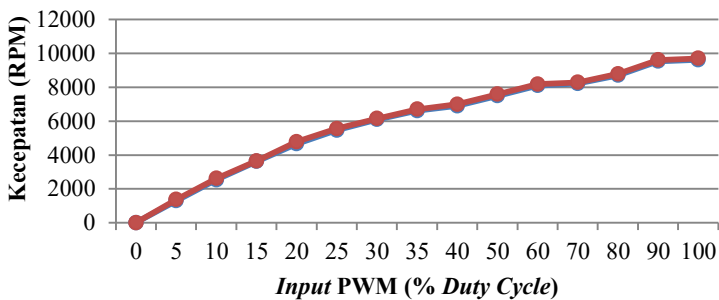
**Gambar 4.4** Potongan Program Rem Magnetik

Kemudian pada Gambar 4.4 merupakan program untuk *driver* rem magnetik. Seperti yang telah dijelaskan pada bab sebelumnya rem magnetik ini dikendalikan kekuatannya dengan menggunakan sinyal PWM.

## 4.2 Pengujian Sensor *Rotary Encoder*

Pada pengujian sensor *rotary encoder*, terdapat 2 pengujian. Yang pertama adalah linierisasi sensor dan yang kedua adalah kalibrasi sensor. Linierisasi sensor dilakukan untuk mengetahui kondisi sensor, apakah

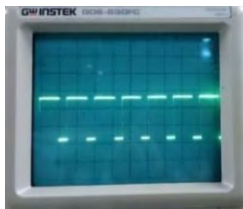
sensor dalam kondisi yang masih bagus atau sudah rusak. Sedangkan kalibrasi sensor dilakukan agar pembacaan kecepatan dari sensor *rotary encoder* akurat. Untuk pengujian linierisasi sensor, caranya yaitu dengan menjalankan motor *brushless* DC dengan sensor *rotary encoder* yang sudah terpasang. Hal ini dilakukan untuk mengetahui respon dan hasil pembacaan kecepatan yang terbaca sensor *rotary encoder* yang dipakai. Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada Lampiran A-5. Gambar 4.5 menunjukkan grafik linierisasi dari sensor *rotary encoder* yang dipakai pada tugas akhir ini.



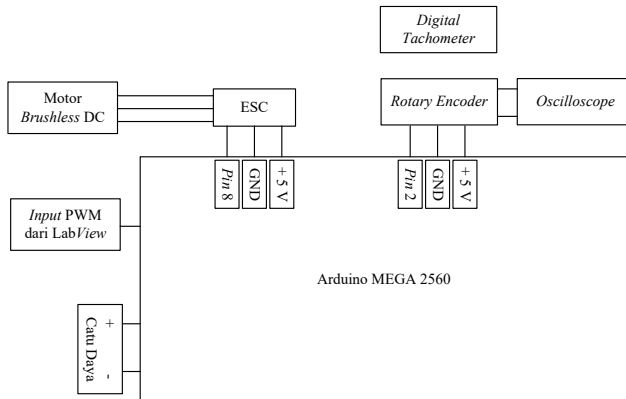
**Gambar 4.5** Linierisasi Sensor *Rotary Encoder*

Dari Gambar 4.5 dapat dilihat bahwa hasil pembacaan sensor *rotary encoder* yang dipakai pada tugas akhir ini sudah cukup linier. Hal ini menunjukkan bahwa sensor masih dalam kondisi yang cukup bagus.

Selain itu, untuk mengetahui kondisi dari sensor *rotary encoder* dapat pula dilakukan pembacaan sinyal pulsa yang dihasilkan oleh sensor melalui osiloskop. Apabila hasil sinyal pulsa berupa kotak sempurna maka sensor dalam kondisi yang bagus. Seperti tertera pada Gambar 4.6 yang merupakan hasil keluaran sinyal pulsa dari sensor yang dipakai pada tugas akhir ini saat kecepatan putaran motor berada pada 3000 RPM.



**Gambar 4.6** Sinyal Pulsa Ketika *Rotary Encoder* Membaca Kecepatan 3000 RPM



**Gambar 4.7** Rangkaian Pengujian Sensor *Rotary Encoder*

**Tabel 4.1** Pengujian Sensor *Rotary Encoder*

<b><i>Input PWM (% Duty Cycle)</i></b>	<b><i>Pembacaan Sensor Rotary Encoder (RPM)</i></b>	<b><i>Pembacaan Sensor Digital Tachometer (RPM)</i></b>	<b><i>Eror (RPM)</i></b>
0	0	0	0
5	1296	1372	76
10	2536	2614	78
15	3616	3652	36
20	4656	4785	129
25	5457	5551	94
30	6097	6155	58
35	6617	6698	81
40	6897	6995	98
50	7497	7590	93
60	8097	8185	88
70	8217	8290	73
80	8697	8787	90
90	9537	9612	75
100	9617	9702	85

Pengujian yang kedua dilakukan untuk mengetahui perbedaan pembacaan antara kecepatan yang diukur oleh sensor *rotary encoder* dari program yang dibuat dengan *digital tachometer* buatan pabrik. Hal

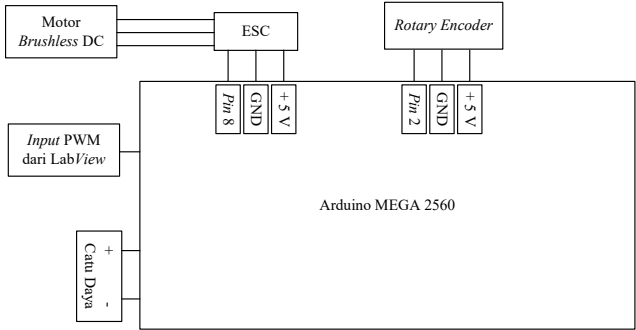
ini dilakukan untuk mengkalibrasi sensor *rotary encoder*. untuk pengujian kalibrasi sensor, caranya yaitu dengan menjalankan motor *brushless* DC dengan sensor *rotary encoder* yang sudah terpasang. Kemudian membandingkan hasil pembacaan kecepatan yang terbaca oleh sensor dengan hasil yang terbaca pada *digital tachometer*. Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada Lampiran A-5. Gambar rangkaian kalibrasi sensor dapat dilihat pada Gambar 4.7. Sedangkan hasil pengkalibrasian sensor *rotary encoder* ini dapat dilihat pada Tabel 4.1.

Pada pengujian yang tertera pada Tabel 4.1 dapat dilihat bahwa *output* hasil pembacaan pada sensor *rotary encoder* dan *digital tachometer* memiliki perbedaan hingga mencapai 100 RPM. Perbedaan kecepatan ini semakin besar pada saat kecepatan tinggi. Hal ini disebabkan karena getaran dari poros motor yang sudah tidak lurus lagi yang menyebabkan getaran piringan *encoder* saat diputar pada kecepatan tinggi.

### 4.3 Pengujian Kecepatan Putaran Motor *Brushless* DC Tanpa Kontroler PID

Pengujian kecepatan motor dilakukan untuk menentukan kemampuan putaran dari motor *brushless* DC yang dipakai. Pengujiannya dilakukan sebanyak tiga kali, yaitu pengujian awal tanpa menggunakan pengereman, pengujian kedua adalah dengan mengkonfigurasi beban motor (rem magnetik) dengan motor yang dipakai dengan setengah pengereman, dan pengujian yang ketiga adalah dengan mengkonfigurasi beban motor (rem magnetik) dengan motor yang dipakai dengan pengereman penuh.

#### 4.3.1 Pengujian Kecepatan Motor Tanpa Pengereman



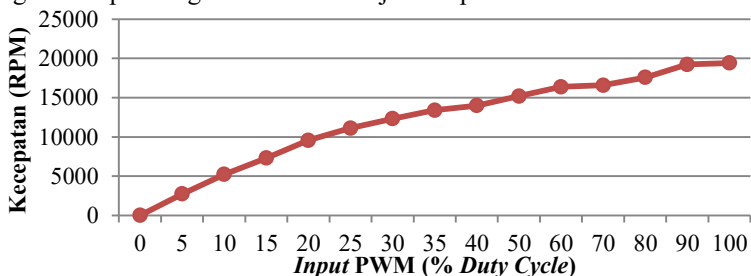
**Gambar 4.8** Rangkaian Pengujian Kecepatan Motor (Tanpa Pengereman)

Pengujian kecepatan motor tanpa pengereman dilakukan untuk mengetahui seberapa cepat motor berputar sesuai dengan *input* yang diberikan. Cara yang dilakukan pada pengujian ini yaitu dengan memberikan *input* dari potensiometer yang nilainya di *mapping* menjadi nilai *throttle* dari 0% sampai dengan 100%. Dengan pengereman yang diberikan adalah sebesar 0%. Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada Lampiran A-5.

**Tabel 4.2** Pengujian Kecepatan Motor (Tanpa Pengereman)

<b><i>Input PWM</i></b> <b>(% Duty Cycle)</b>	<b>Kecepatan Motor</b> <b>(RPM)</b>
0	0
5	2744
10	5228
15	7304
20	9570
25	11102
30	12310
35	13396
40	13990
50	15180
60	16370
70	16580
80	17574
90	19224
100	19404

Tabel 4.2 menunjukkan hasil dari pengujian kecepatan motor *brushless* DC tanpa menggunakan pengereman elektromagnetik. Sedangkan grafik dari kecepatan putaran motor *brushless* DC yang digunakan pada tugas akhir ini ditunjukkan pada Gambar 4.9.



**Gambar 4.9** Hasil Pengujian Kecepatan Motor (Tanpa Pengereman)

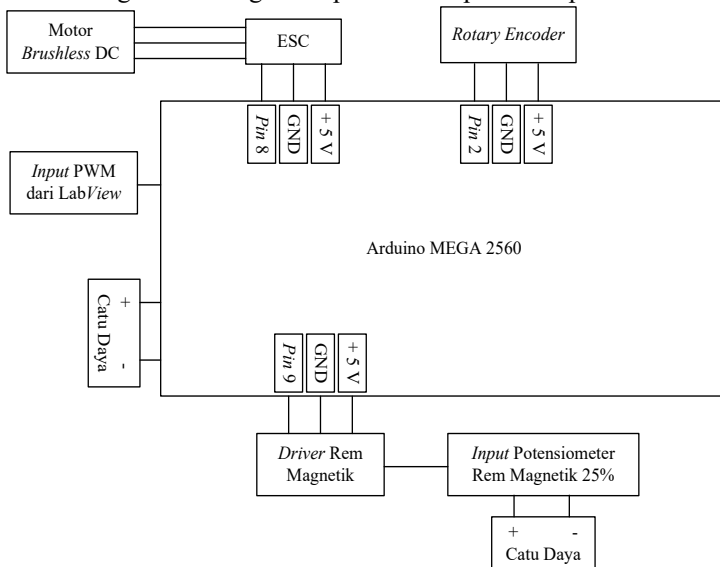
Berdasarkan pengujian yang telah dilakukan dengan hasil yang didapat sesuai Tabel 4.2 diatas, maka didapatkan kesimpulan bahwa pengaturan kecepatan motor *brushless* DC dapat dilakukan dengan mengatur besar masukan PWM-nya.

#### 4.3.2 Pengujian Kecepatan Motor Dengan Setengah Pengereman

Pada pengujian kecepatan motor dengan setengah pengereman, pengukuran kecepatan motor *brushless* DC dikonfigurasi dengan beban motor (rem magnetik). Pengujian ini dilakukan ketika rem magnetik berada dalam kondisi setengah pengereman.

Pengujian ini dilakukan dengan menggunakan *input* dari potensiometer yang nilainya di *mapping* menjadi nilai *throttle* dari 0% sampai 100%. Pada Tabel 4.3 menunjukkan hasil dari pengujian kecepatan motor *brushless* DC dengan menggunakan pengereman elektromagnetik setengah pengereman.

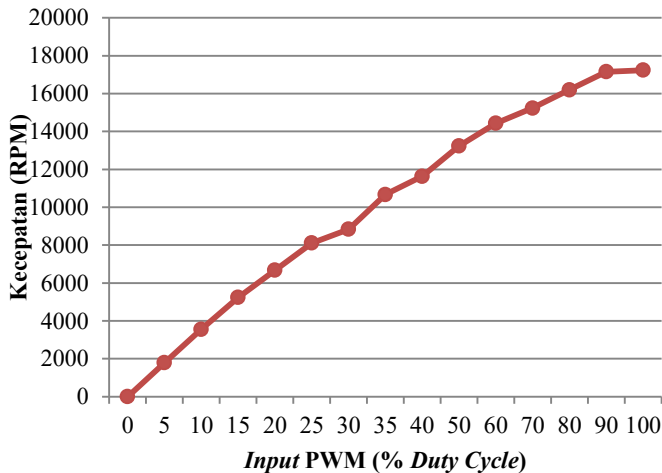
Cara yang digunakan agar pengereman bisa mencapai setengah pengereman adalah dengan memutar potensiometer yang telah dipasang pada *driver* rem magnetik. Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada Lampiran A-5.



**Gambar 4.10** Rangkaian Pengujian Kecepatan Motor (Dengan Setengah Pengereman)

**Tabel 4.3** Pengujian Kecepatan Motor (Dengan Setengah Pengereman)

<i>Input PWM (% Duty Cycle)</i>	<i>Kecepatan Motor (RPM)</i>
0	0
5	1792
10	3552
15	5232
20	6672
25	8112
30	8832
35	10674
40	11634
50	13234
60	14434
70	15234
80	16194
90	17154
100	17234



**Gambar 4.11** Hasil Pengujian Kecepatan Motor (Dengan Setengah Pengereman)

Pengujian kecepatan motor dengan setengah pengereman dilakukan untuk menentukan kemampuan putaran dari motor *brushless*



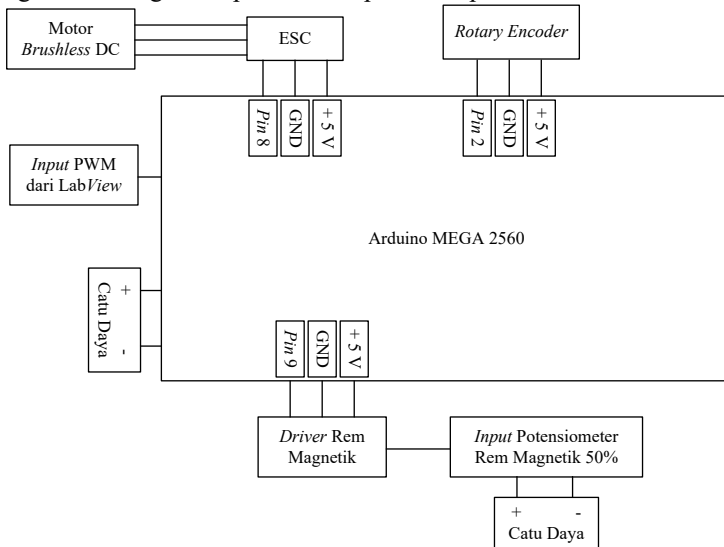
DC yang dipakai apabila motor diberi beban. Pengujiannya dilakukan ketika motor diberikan beban rem magnetik yang dikopel dengan motor *brushless* DC. Akibat dari pembebanan tersebut maka kecepatan motor berkurang. Kecepatan motor dengan setengah pengereman ditunjukkan pada Tabel 4.3 dengan hasil grafik kecepatan yang ditunjukkan pada Gambar 4.11.

#### 4.3.3 Pengujian Kecepatan Motor Dengan Pengereman Penuh

Pada pengujian kecepatan motor dengan pengereman penuh, pengukuran kecepatan motor *brushless* DC dikonfigurasi dengan beban motor (rem magnetik). Pengujian ini dilakukan dengan pengereman penuh.

Pengujian ini dilakukan dengan menggunakan *input* dari potensiometer yang nilainya di *mapping* menjadi nilai *throttle* dari 0% sampai 100%. Pada Tabel 4.4 akan ditunjukkan hasil dari pengujian kecepatan motor *brushless* DC dengan menggunakan pengereman elektromagnetik pengereman penuh.

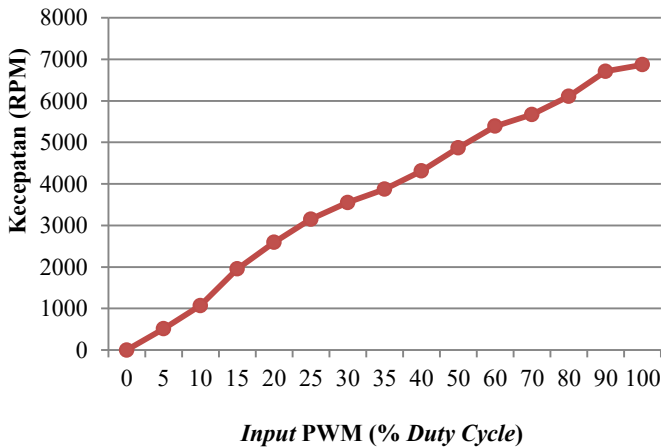
Cara yang digunakan agar pengereman bisa mencapai pengereman penuh adalah dengan memutar potensiometer yang telah dipasang pada *driver* rem magnetik. Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada Lampiran A-5.



**Gambar 4.12** Rangkaian Pengujian Kecepatan Motor (Dengan Pengereman Penuh)

**Tabel 4.4** Pengujian Kecepatan Motor (Dengan Pengereman Penuh)

<b><i>Input PWM (% Duty Cycle)</i></b>	<b><i>Kecepatan Motor (RPM)</i></b>
0	0
5	512
10	1072
15	1952
20	2592
25	3152
30	3552
35	3872
40	4312
50	4872
60	5392
70	5672
80	6112
90	6712
100	6872



**Gambar 4.13** Hasil Pengujian Kecepatan Motor (Dengan Pengereman Penuh)

Pengujian kecepatan motor dengan pengereman penuh dilakukan untuk menentukan kemampuan putaran dari motor *brushless* DC yang dipakai apabilaa diberi beban penuh. Pengujiannya dilakukan ketika motor diberikan beban rem magnetik yang dikopel dengan motor

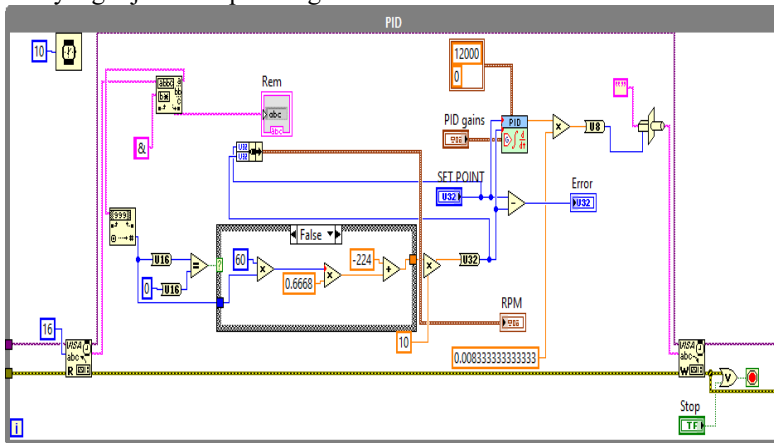
*brushless* DC. Akibat dari pembebanan tersebut maka kecepatan motor berkurang. Kecepatan motor dengan setengah pengereman ditunjukkan pada Tabel 4.4 dengan hasil grafik kecepatan yang ditunjukkan pada Gambar 4.13.

#### 4.4 Pengujian *Software*

Pengujian *software* dilakukan untuk memastikan program-program perangkat lunak yang sudah dibuat dapat bekerja sesuai dengan yang diharapkan dan dapat bersinergi dengan *hardware* untuk menjalankan sistem sesuai dengan yang direncanakan.

#### 4.4.1 Kontroler PID LabView

Pada Gambar 4.14 menunjukkan bagaimana model dari kontroler PID yang dijalankan pada tugas akhir ini.



**Gambar 4.14** Blok Diagram Kontroler PID LabView

#### 4.4.2 Pengujian Motor *Brushless* DC Dengan Kontroler PID

Pengujian motor *brushless* DC dengan kontroler PID ini dimaksudkan untuk menunjukkan apakah alat sudah berjalan tanpa gangguan dan berjalan sesuai fungsinya. Maksudnya adalah ketika diberikan kontroler PID, maka seberapa besar pun pengereman yang diberikan pada motor, *output* kecepatan yang dihasilkan akan tetap sama dengan *input* kecepatan yang diberikan.

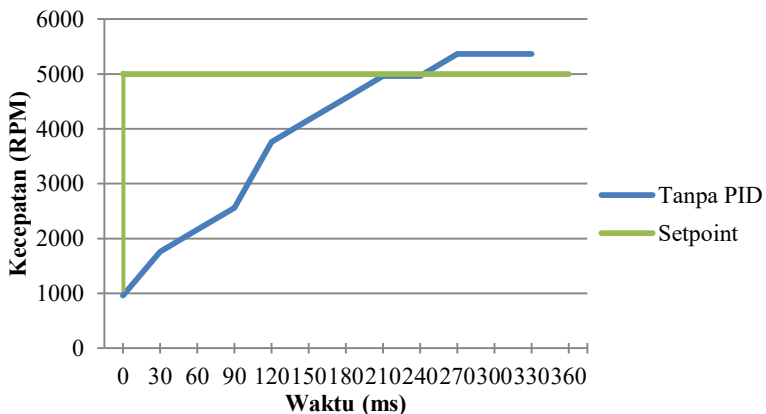
Pengujian ini dilakukan dengan melihat respon yang diberikan oleh motor *brushless* DC ketika motor *brushless* DC diberi *input* dari *set point* yang telah ditentukan sekaligus dengan pembebanannya (rem

magnetik). Pengujian ini juga memperlihatkan bagaimana pengaruh kontroler PID pada kecepatan putaran motor *brushless* DC.

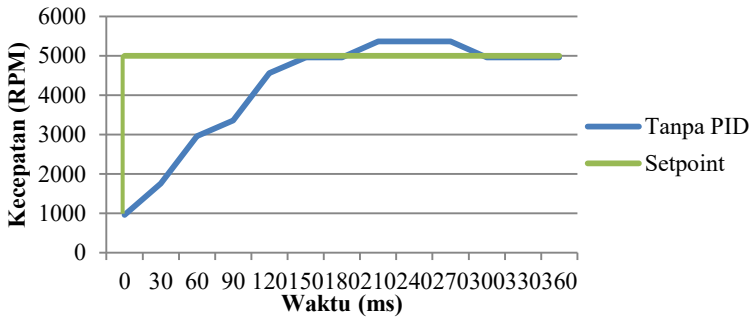
Pada tugas akhir ini, motor *brushless* DC akan dikopel dengan rem magnetik dengan *rotary encoder* yang berfungsi untuk mengukur seberapa banyak putaran yang dihasilkan oleh motor tersebut dalam suatu waktu. Banyaknya putaran yang dihasilkan oleh motor *brushless* DC dan berapa besarnya pengereman yang dilakukan tersebut akan ditampilkan pada LabView.

Oleh karena tugas akhir ini nantinya akan dijadikan sebagai modul praktikum, maka selain ditampilkan pada LabView yang letaknya ada pada praktikan, banyaknya putaran yang dihasilkan oleh motor *brushless* DC dan berapa besarnya pengereman yang dilakukan juga akan dikirimkan ke komputer lain. Komputer lain ini adalah komputer yang dipakai oleh asisten laboratorium sehingga asisten laboratorium bisa memonitor hasil-hasil yang didapat oleh praktikan. Pengiriman data dari komputer praktikan dengan komputer asisten laboratorium dilakukan dengan media *wireless* Ad Hoc. Gambar keseluruhan blok diagram dari LabView tertera pada Lampiran A-5.

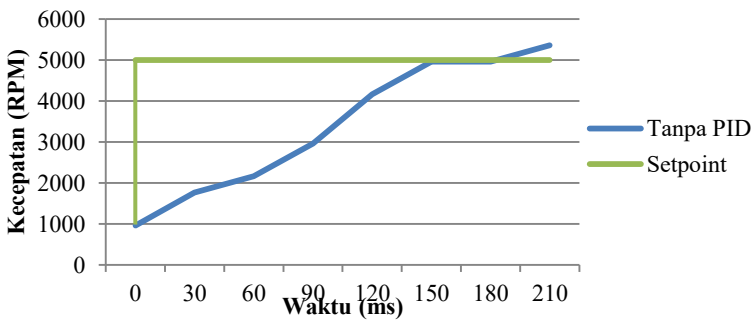
Pada Gambar 4.15, Gambar 4.16, dan Gambar 4.17 menunjukkan grafik respon kecepatan motor *brushless* DC sebelum menggunakan kontroler PID dengan pengereman yang berbeda. Sehingga didapatkanlah Tabel 4.5 yang menunjukkan kecepatan motor ketika tanpa kontroler PID. Tabel ini kemudian menjadi acuan dalam pencarian parameter kontroler PID dengan menggunakan metode *trial and error*.



**Gambar 4.15** Kecepatan Motor *Brushless* DC Pengereman 0% Tanpa Kontroler PID



**Gambar 4.16** Kecepatan Motor *Brushless* DC Pengereman 25% Tanpa Kontroler PID



**Gambar 4.17** Kecepatan Motor *Brushless* DC Pengereman 50% Tanpa Kontroler PID

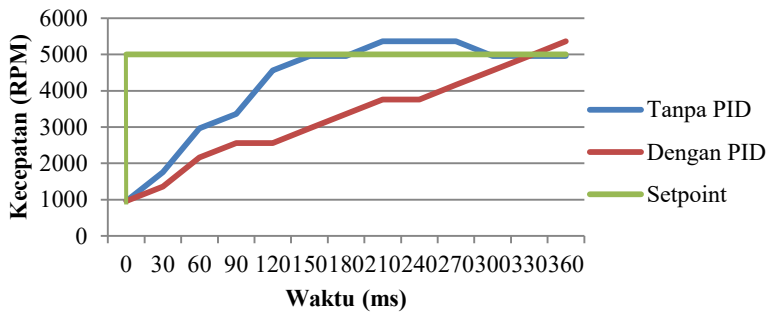
**Tabel 4.5**Data Kecepatan Motor Tanpa Kontroler PID

Pengereman (%)	Settling Time (Ts)	Error Steady State (Ess)	Rise Time (Tr)
0%	317 ms	39 RPM	196 ms
25%	171 ms	39 RPM	116 ms
50%	192 ms	39 RPM	118 ms

Tabel 4.6 dan Tabel 4.7 merupakan sebagian dari *history* pencarian parameter kontroler PID dengan metode *trial and error*. Grafik kecepatan motor ditunjukkan pada Gambar 4.18 dan Gambar 4.19 dengan pengereman sebesar 25%.

Tabel 4.6 History Perubahan Nilai  $K_p$ ,  $T_i$ ,  $T_d$  Pada Setiap Pengereman

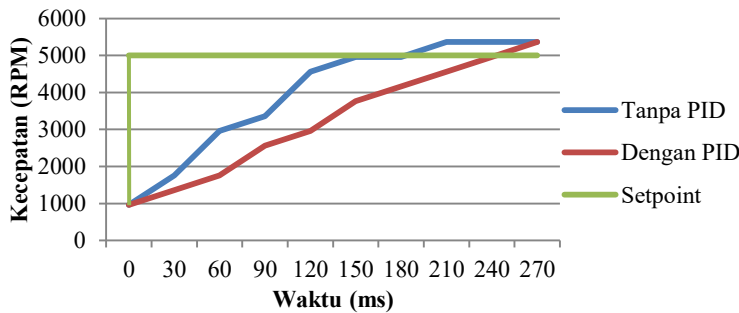
Pengereman	$K_p$	$T_i$	$T_d$
Tanpa Pengereman (0%)	0,178	0,034	0,000
Setengah Pengereman (25%)	0,178	0,036	0,005
Pengereman Penuh (50%)	0,178	0,040	0,005



Gambar 4.18 History Perbandingan Kecepatan Motor Brushless DC Pengereman 25%

Tabel 4.7History Perubahan Nilai  $K_p$ ,  $T_i$ ,  $T_d$  Pada Setiap Pengereman

Pengereman	$K_p$	$T_i$	$T_d$
Tanpa Pengereman (0%)	0,115	0,009	0,000
Setengah Pengereman (25%)	0,099	0,006	0,000
Pengereman Penuh (50%)	0,098	0,003	0,000



Gambar 4.19 History Perbandingan Kecepatan Motor Brushless DC Pengereman 25%

Dengan beberapa *history* parameter kontroler PID seperti ditunjukkan pada Tabel 4.6 dan Tabel 4.7 serta melihat grafik kecepatan motor seperti ditunjukkan pada Gambar 4.18 dan 4.19, maka dipilih parameter terbaik seperti ditunjukkan pada Tabel 4.8 dengan hasil kecepatan motor seperti ditunjukkan pada Tabel 4.9.

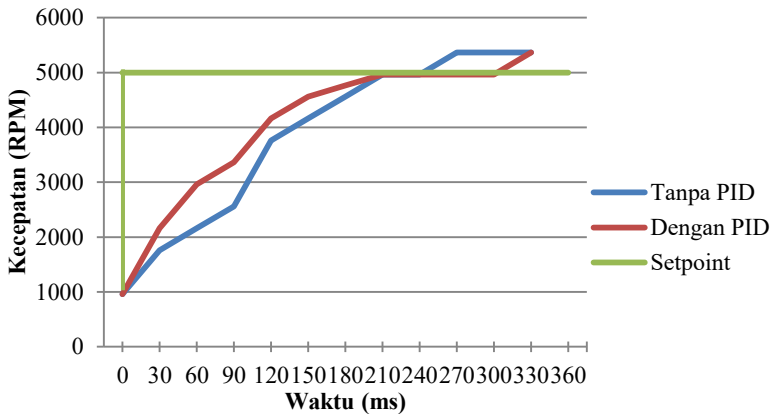
**Tabel 4.8**Perubahan Nilai  $K_p$ ,  $T_i$ ,  $T_d$  Pada Setiap Pengereman

Pengereman	$K_p$	$T_i$	$T_d$
Tanpa Pengereman (0%)	0,100	0,006	0,000
Setengah Pengereman (25%)	0,085	0,003	0,010
Pengereman Penuh (50%)	0,054	0,002	0,020

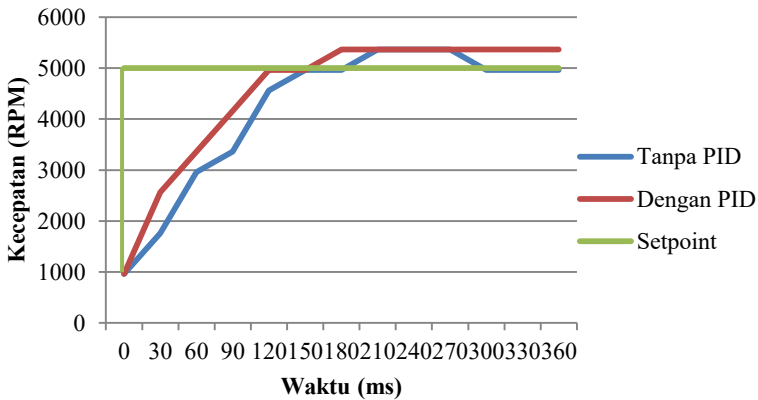
**Tabel 4.9**Data Kecepatan Motor Dengan Kontroler PID

Pengereman (%)	Settling Time (Ts)	Error Steady State (Ess)	Rise Time (Tr)
0%	234 ms	39 RPM	185 ms
25%	141 ms	139 RPM	105 ms
50%	189 ms	39 RPM	122 ms

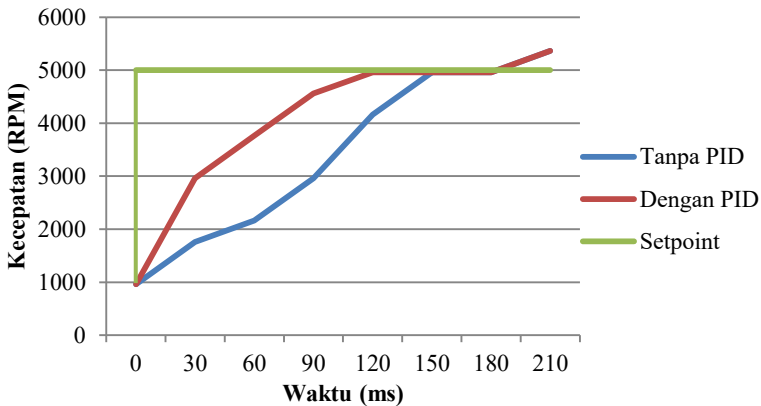
Gambar 4.20, Gambar 4.21, dan Gambar 4.22 menunjukkan grafik kecepatan motor *brushless* DC yang sudah menggunakan parameter kontroler PID terbaik dari *history trial and error* yang sudah dilakukan sebelumnya.



**Gambar 4.20** Perbandingan Kecepatan Motor *Brushless* DC Pengereman 0%



**Gambar 4.21** Perbandingan Kecepatan Motor *Brushless* DC Pengereman 25%



**Gambar 4.22** Perbandingan Kecepatan Motor *Brushless* DC Pengereman 50%

Setelah ditemukan parameter kontroler PID yang terbaik. Kemudian dilakukan pengujian pada kecepatan motor *brushless* DC. pengujian ini dilakukan untuk membuktikan bahwa *output* kecepatan nantinya akan sama dengan *input* atau *setpoint* yang diberikan walaupun diberikan pengereman yang berbeda-beda. Yaitu tanpa pengereman (pengereman 0%), pengereman setengah (pengereman 25%), dan pengereman penuh (pengereman 50%). Program dari pengujian ini menggunakan LabView dengan blok diagram seperti tertera pada



Lampiran A-5. Dengan hasil dari pengujian seperti ditunjukkan pada Tabel 4.10.

**Tabel 4.10** Hasil Kecepatan Motor *Brushless* DC Dengan Kontroler PID

<i>Set Point</i> (RPM)	Pengereman (%)	Kecepatan Motor (RPM)
2000	0	2161
3000	0	2961
4000	0	4161
5000	0	4961
6000	0	5762
2000	25	2161
3000	25	2961
4000	25	4161
5000	25	4961
6000	25	5762
2000	50	2161
3000	50	2961
4000	50	4161
5000	50	4961
6000	50	5762

## **LAMPIRAN A**

### ***LISTING PROGRAM***

#### ***LISTING PROGRAM ARDUINO MEGA 2560***

```
volatile byte half_revolutions;
#include <Servo.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(22, 24, 26, 28, 30, 32);
#include <TimerOne.h>
unsigned int counter = 0;
int rotation, rotation1;
float xxx;
int sensorPin = A1; // input analog potentiometer
unsigned long offset;

int sensorValue ; // variable integer untuk menampung nilai
potensiometer
unsigned long timeold;
Servo myservo;
unsigned int rpm;

//Min & max pulse
//int minPulse = 1000;
int maxPulse = 2000;

int speed = 0;

void setup()
{
  xxx = 0.0;
  Serial.begin(115200);
  arus = 0;
  lcd.begin(16, 2);
  pinMode(2, INPUT);
  Timer1.initialize(100000);
  attachInterrupt(0, docount, RISING);
  Timer1.attachInterrupt( timerIsr );
```

```

myservo.attach(8); // pin 8 sebagai output ke ESC
half_revolutions = 0;
rpm = 0;
timeold = 0;
x = 0;
myservo.write(x);
for (i = 0; i < 1000; i++)

{
    offset = offset + (analogRead(A0));
    delay(1);
}

offset = 512 - (offset / 1000);
//rotation = 0;
}

void loop()
{
    char pp[1024] = {0};
    char y[12];
    char z[10];
    sprintf(y, "%d", rotation);
    sprintf(z, "%d", sensorValue);
    itoa(rotation, y, 10);
    strcat(pp, y);
    strcat(pp, "&");
    itoa(sensorValue, z, 10);
    strcat(pp, z);
    Serial.println(pp);
    xxx = (rotation * 60 * 0.6668) - 224;
    rotation1 = int(xxx);
    if (rotation1 < 0) {
        rotation1 = 0;
    }

    sensorValue = analogRead(sensorPin);
    //delay(10);
    sensorValue = map(sensorValue, 0, 1023, 0, 255);

```

```

analogWrite(9, sensorValue);
sensorValue = map(sensorValue, 0, 255, 0, 100);

if (Serial.available() > 0)

{
  x = Serial.read();
  if (x < 0) {
    x = 0;
  }
  myservo.write(x);

}
//Serial.print(x);
//Serial.print(" ");
lcd.setCursor(0, 0);
lcd.print(rotation1);
lcd.setCursor(0, 1);
lcd.print(sensorValue);
delay(10);
lcd.clear();
}

/*void rpm_fun()
{
  half_revolutions++;
  //Each rotation, this interrupt function is run twice
}*/

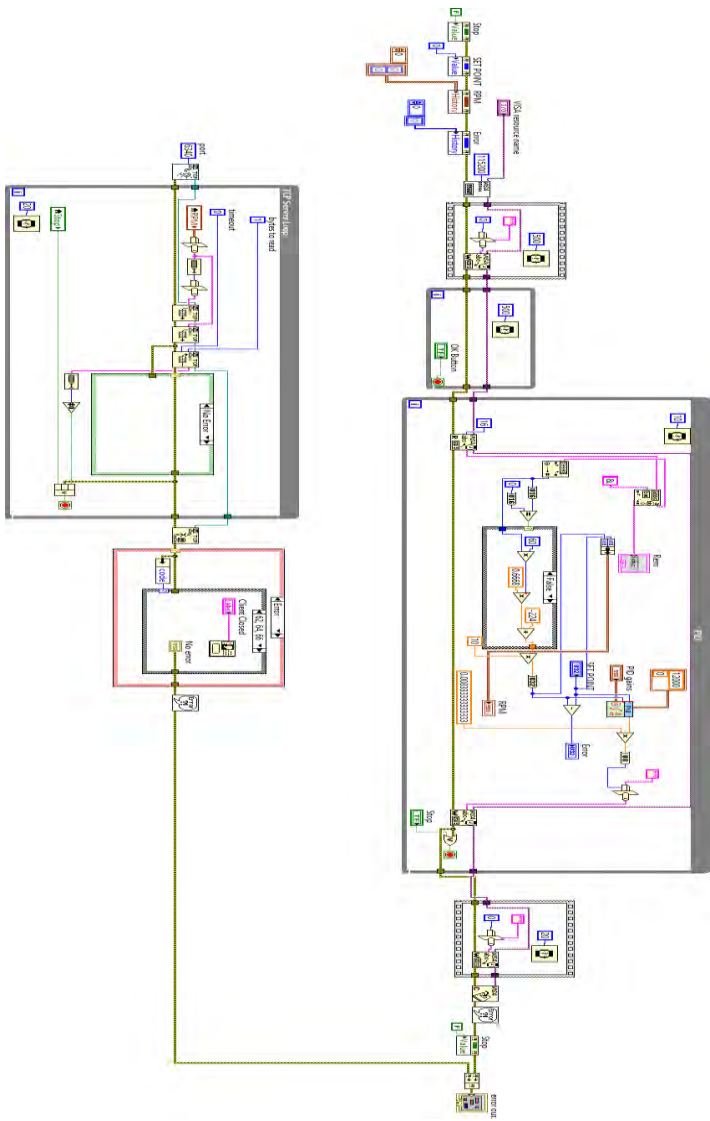
void docount()
{
  counter++;
}

void timerIsr()
{
  // Serial.print("Motor Speed: ");
  counter = counter / 4;
  rotation = counter;
}

```

```
//Serial.println(rotation);  
// Serial.println(" Rotation per second");  
counter = 0;  
  
}
```

**BLOK DIAGRAM LABVIEW**



*-- Halaman ini sengaja dikosongkan --*

## LAMPIRAN B *DATASHEET*

### 1. *Datasheet* Arduino Mega 2560



#### Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

#### Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- ✦ **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- ✦ **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.



✦ **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

✦ **GND.** Ground pins.

## Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the [EEPROM library](#)).

## Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

✦ **Serial:** **0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

✦ **External Interrupts:** **2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt\(\)](#) function for details.

✦ **PWM:** **0 to 13.** Provide 8-bit PWM output with the `analogWrite()` function.

✦ **SPI:** **50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the [SPI library](#). The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.

✦ **LED:** **13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

✦ **I<sup>2</sup>C:** **20 (SDA) and 21 (SCL).** Support I<sup>2</sup>C (TWI) communication using the [Wire library](#) (documentation on the [Wiring website](#)). Note that these pins are not in the same location as the I<sup>2</sup>C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and `analogReference()` function.

There are a couple of other pins on the board:

✦ **AREF.** Reference voltage for the analog inputs. Used with `analogReference()`.

✦ **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A [SoftwareSerial library](#) allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I<sup>2</sup>C (TWI) and SPI communication. The Arduino software includes a [Wire library](#) to simplify use of the I<sup>2</sup>C bus; see the [documentation on the Wiring website](#) for details. For SPI communication, use the [SPI library](#).

## Programming

The Arduino Mega can be programmed with the Arduino software ([download](#)). For details, see the [reference](#) and [tutorials](#).

The ATmega2560 on the Arduino Mega comes preburned with a [bootloader](#) that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](#), [C header files](#)).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see [these instructions](#) for details.

The ATmega8U2 firmware source code is available in the [Arduino repository](#). The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use [Atmel's FLIP software](#) (Windows) or the [DFU programmer](#) (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See [this user-contributed tutorial](#) for more information.

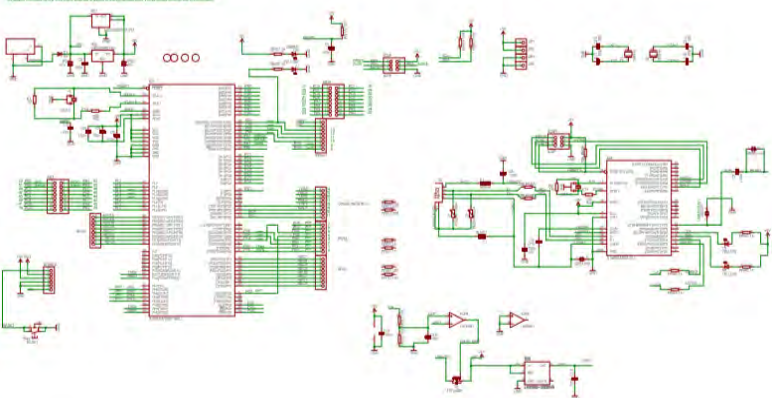
## Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. Please note that I<sup>2</sup>C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).

### Arduino® Mega 2560 Reference Design

THIS DOCUMENT IS PROVIDED AS A REFERENCE DESIGN. IT IS NOT A WARRANTY, REPRESENTATION, OR GUARANTEE OF ANY KIND. THE USER ASSUMES ALL LIABILITY FOR THE USE OF THIS DOCUMENT. THE USER SHALL INDEMNIFY AND HOLD HARMLESS ATMELO CORPORATION AND ITS AFFILIATES FROM AND AGAINST ALL CLAIMS, DAMAGES, LOSSES, AND EXPENSES, INCLUDING REASONABLE ATTORNEY'S FEES, ARISING OUT OF OR RESULTING FROM THE USE OF THIS DOCUMENT.



## 2. Datasheet Driver Motor RCTimer ESC 30 A

### Manual of RC Timer ESC 30A Brushless Motor Speed Controller

Thanks for purchasing RC Timer Electronic Speed Controller (ESC). High power system for RC model can be very dangerous, so we strongly suggest you read this manual carefully, in that we have no control over the correct use, installation, application, or maintenance of our products, no liability shall be assumed nor accepted for any damages, losses or costs resulting from the use of the product. Any claims arising from the operating, failure or malfunctioning etc. will be denied. We assume no liability for personal injury, property damage or consequential damages resulting from our product or our workmanship. As far as is legally permitted, the obligation to compensation is limited to the invoice amount of the affected product.

#### Specification

Input voltage: DC 6-16.8V(2-4S Lipo)  
 Running current:30A(Output: Continuous 30A, Burst 40A up to 10 Secs )  
 Size: 28mm (L) \* 23mm (W) \* 11mm (H)  
 Weight: 32g

#### Features

- Extreme low output resistance, super current endurance.
- Multiple protection features: Low-voltage cut-off protection / over-heat protection / throttle signal loss protection.
- 3 start modes: Normal / Soft / Super-Soft, compatible with fixed-wing aircraft and helicopter.
- Throttle range can be configured to be compatible with all transmitters currently available on market.
- Smooth, linear and precise throttle response.
- Separate voltage regulator IC for microprocessor (except Pentium-8A and Pentium-10A), providing good anti-jamming capability.
- Supported motor speed (Maximum): 21000 RPM (2 poles), 7000 RPM (6 poles), 3500 RPM (12 poles).

#### Wire Diagram



**Very important:** If you use banana-strap connectors on main power wires (input wires), please connect the black wire (negative polarity) BEFORE the red wire (positive polarity) so the right

sequence is: BOMP Adapter → BLACK wire of main power → RED wire of main power

#### Programmable Items:

1. **Brake Setting:** Enabled / Disabled, default is Disabled
2. **Battery Type:** Li-xx(Li-Ion or Li-poly) / Ni-xx(NiMH or NiCd), default is Li-xx.
3. **Low Voltage Protection Mode(Cut-Off Mode):** Soft Cut-Off (Gradually reduce the output power) or Cut-Off (Immediately stop the output power). Default is Soft Cut-Off.
4. **Low Voltage Protection Threshold(Cut-Off Threshold):** Low / Medium / High, default is Medium.  
 1) For lithium batteries, the number of battery cells is calculated automatically. Low / medium / high output voltage for each cell is: 2.6V/2.55V/3.1V. For example: For a 3 cells lithium pack, when "Medium" cutoff threshold is set, the cut-off voltage will be: 2.55\*3=3.85V.  
 2) For nickel batteries, low / medium / high cutoff voltages are 0%/45%/80% of the startup voltage (i.e. the initial voltage of battery pack), and 0% means the low voltage out-of function is disabled. For example: For a 10 cells NiMH battery, fully charged voltage is 1.44\*6=8.64V, when "Medium" cut-off threshold is set, the cut-off voltage will be: 8.64\*50%=4.32V.
5. **Startup Mode:** Normal / Soft / Super-Soft, default is Normal.  
 Normal is preferred for fixed-wing aircraft. Soft or Super-Soft are preferred for helicopters. The initial acceleration of the Soft and Super-Soft modes are slower in comparison, usually taking 1 second for Soft startup or 2 seconds for Super-Soft startup from initial throttle advance to full throttle. If the throttle is closed (throttle stick moved to bottom) and opened again (throttle stick moved to top) within 3 seconds of the initial startup, the restart-up will be temporarily changed to normal mode to get rid of the chances of a crash caused by slow throttle response. This special design is very suitable for aerobatic flight when quick throttle response is needed.
6. **Timing:** Low / Medium / High, default is Low.  
 Usually, low timing value can be used for most motors. We recommend the Low timing value for 2 poles motor and Medium timing value for motors with more than 6 poles to get a high efficiency. For higher speed, high timing value can be chosen.

#### Special Note

Some high KV out-runner motors have very special construction, the space between each magnet is very large, and many ESCs can't drive these motors. After much testing, our ESCs have proven to work very well with these types of motors. Therefore, we have provided some suggestions as follows:

Motor	Programmable	Timing	Startup mode
Generic In-runner motor	Low		Usually aircraft use "Normal" startup mode and helicopter use "Super-soft" startup mode
Generic out-runner motor	Low or Medium		
Align 420LF (Made in TAIWAN, out-runner)	High (MUST)		
480TH (Made in TAIWAN)	Low		Soft/MUST

#### Begin To Use Your New ESC

Please start the ESC in the following sequences:

1. Move the throttle stick to the bottom position and then switch on the transmitter.
2. Connect the battery pack to the ESC, the ESC begins the self-test process, a special tone "123" is emitted, which means the voltage of the battery pack is in normal range, and then "N" "beep" tones will be emitted, means the number of lithium battery cells. Finally a long "beep———" tone will be emitted, which means self-test is OK, the aircraft.  
 □ If a special tone "56712" is emitted after 2 beep tones ("beep-beep-"), means the ESC has entered the program mode, it is because the throttle channel of your transmitter is reversed, please set it correctly.

## Program the ESC with your transmitter (4 Steps):

1. Enter program mode
2. Select programmable items
3. Set item's value (Programmable value)
4. Exit program mode

**1. Enter program mode**  
 1) Switch on transmitter, move throttle stick to top, connect the battery pack to ESC.  
 2) Wait for 2 seconds, the motor should emit special tone like "beep-beep-"  
 3) Wait for another 5 seconds, special tone like "S6715" should be emitted, which means program mode is entered.



- 2. Select programmable items:**  
 After entering program mode, you will hear 3 tones in a loop with the following sequence. If you move the throttle stick to bottom within 3 seconds after one kind of tones, this item will be selected.
1. "beep" brake; (1 short tone)
  2. "beep-beep-" battery type (2 short tones)
  3. "beep-beep-beep-" cutoff mode (3 short tones)
  4. "beep-beep-beep-beep-" cutoff threshold (4 short tones)
  5. "beep-—" startup mode (1 long tone)
  6. "beep-—" timing (1 long 1 short)
  7. "beep-—"beep-beep-" set all to default (1 long 2 short)
  8. "beep-—"beep-—" exit (2 long tone)
- Note: 1 long "beep-—" = 5 short "beep-"

### 3. Set item value (Programmable value):

You will hear several tones in loop. Set the value matching to a tone by moving throttle stick to top when you hear the tone, then a special tone "1515" emits, means the value is set and saved. (Keeping the throttle stick at top, you will go back to step 2 and you can select other items. Moving the stick to bottom within 2 seconds will exit program mode directly).

Tone's	"beep-" 1 short tone	"beep-beep-" 2 short tones	"beep-beep-beep-" 3 short tones
Brake	off	on	
Battery type	Lipo / Li-poly	NiMH / NiCd	
Cutoff mode	Soft-Cut	Clap-off	
Cutoff threshold	Low	Medium	High
Startup mode	Normal	Soft	Super soft
Timing	Low	Medium	High

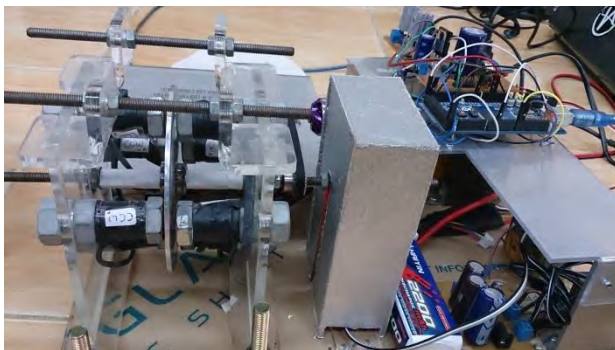
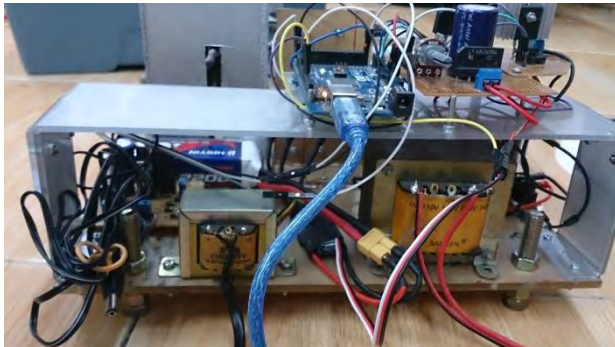
### 4. Exit program mode

There are 2 ways to exit program mode:

1. In step 3, after special tone "1515", please move throttle stick to the bottom position, within 2 seconds.
2. In step 2, after tone "beep-—"beep-—" (ie. The item #8), move throttle stick to bottom within 3 seconds.

*-- Halaman ini sengaja dikosongkan --*

## LAMPIRAN C *DOKUMENTASI*



*-- Halaman ini sengaja dikosongkan --*

## BAB V PENUTUP

Bab penutup ini berisi tentang kesimpulan yang diperoleh selama proses pembuatan modul kontrol kecepatan motor *brushless* DC, kesimpulan dari hasil pengujian dan analisa data, serta saran untuk modul kontrol kecepatan motor *brushless* DC ini kedepannya.

### 5.1 Kesimpulan

1. Sensor *rotary encoder* yang digunakan dalam kondisi yang cukup bagus dengan eror pembacaan kurang dari 5%.
2. Hasil pengujian kecepatan putaran motor sebelum menggunakan kontroler PID pada *input* PWM 10% *duty cycle* ketika pengereman 0% adalah sebesar 5228 RPM, pengereman 25% adalah sebesar 3552 RPM, dan pengereman 50% adalah sebesar 1072 RPM dengan eror pembacaan kurang dari 5%.
3. Hasil pengujian kecepatan putaran motor setelah menggunakan kontroler PID akan menghasilkan kecepatan yang sama dengan *setpoint* kecepatan yang diberikan. Misalnya dalam tugas akhir ini pada *setpoint* 2000 RPM ketika pengereman 0% adalah sebesar 2161 RPM, pengereman 25% adalah sebesar 2161 RPM, dan pengereman 50% adalah sebesar 2161 RPM dengan eror pembacaan kurang dari 5%.
4. Parameter kontroler PID yang digunakan pada tugas akhir ini yaitu  $K_p=0,100$   $T_i=0,006$   $T_d=0,000$  pada pengereman 0%,  $K_p=0,085$   $T_i=0,003$   $T_d=0,010$  pada pengereman 25%, dan  $K_p=0,054$   $T_i=0,002$   $T_d=0,020$  pada pengereman 50%.

### 5.1 Saran

1. Diharapkan kedepannya dapat memilih komponen-komponen yang dibutuhkan dengan lebih tepat. Karena selama perancangan alat yang lalu masih sering terjadi alat berfungsi tidak maksimal dikarenakan komponen penyusunnya kurang tepat.
2. Keseluruhan alat baik penyangga maupun tatanan lebih baik dibuat dari bahan yang kuat dan tahan terhadap getaran, agar sewaktu dijalankan tidak mengganggu kecepatan putar.



***-- Halaman ini sengaja dikosongkan --***

## DAFTAR PUSTAKA

- [1] Hudaibiy Hibban. “*Desain Kontroler Fuzzy PID Gain Schedulling Untuk Pengaturan Kecepatan Motor DC Tanpa Sikat*”, **Tugas Akhir**, Jurusan Teknik Elektro FTI-ITS, Surabaya, 2015.
- [2] Bachtiar, Muhammad Fachri, Priyatna, Alif Gigah. “*Perancangan Rem Magnetik Pada Motor DC Dengan Menggunakan Arduino*”, **Tugas Akhir**, D3 Teknik Elektro, ITS Surabaya, 2015.
- [3] A. Trivino-Cabrera. “**Guide to Wireless Ad Hoc Networks**”, Springer, USA, 2009.
- [4] Mihura, Bruce. “**Learning LabView For The First Time**”, National Instruments, Washington, 2001.
- [5] Ogata, Katsuhiko. “**Teknik Kontrol Automatik (Sistem Pengaturan) Jilid 1**”, Penerbit Erlangga, Jakarta, 1991.
- [6] Pramudijanto, Jos. **Catatan Kuliah Penulisan Ilmiah**, <http://www.ee.its.ac.id/~jos>, Jurusan Teknik Elektro ITS, 10 Januari 2016.

*-- Halaman ini sengaja dikosongkan --*



*-- Halaman ini sengaja dikosongkan --*

